

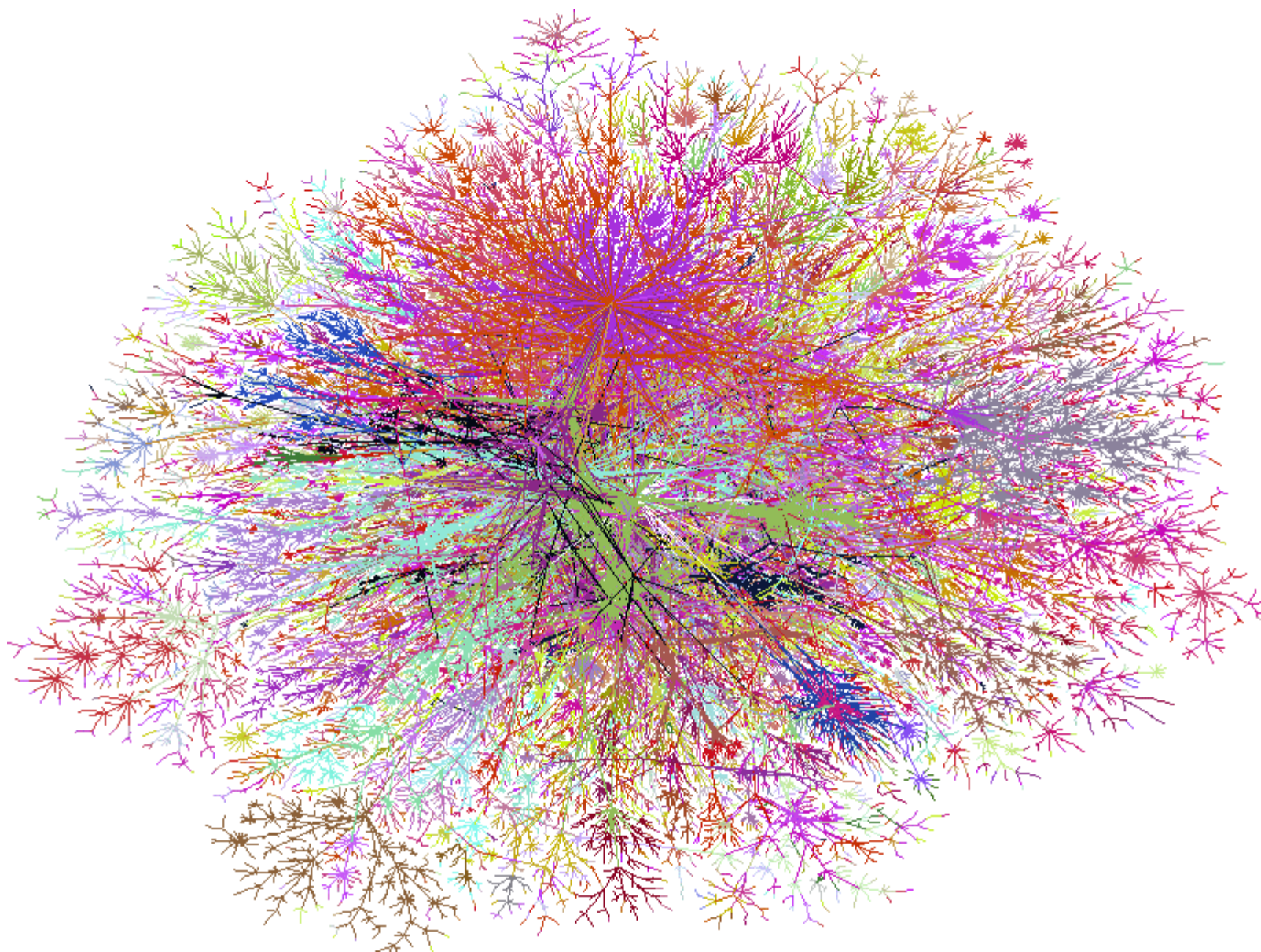
# **Sissejuhatus infotehnoloogiasse**

Interneti funktsioneerimine

# Loengu ülevaade. Interneti funktsioneerimine

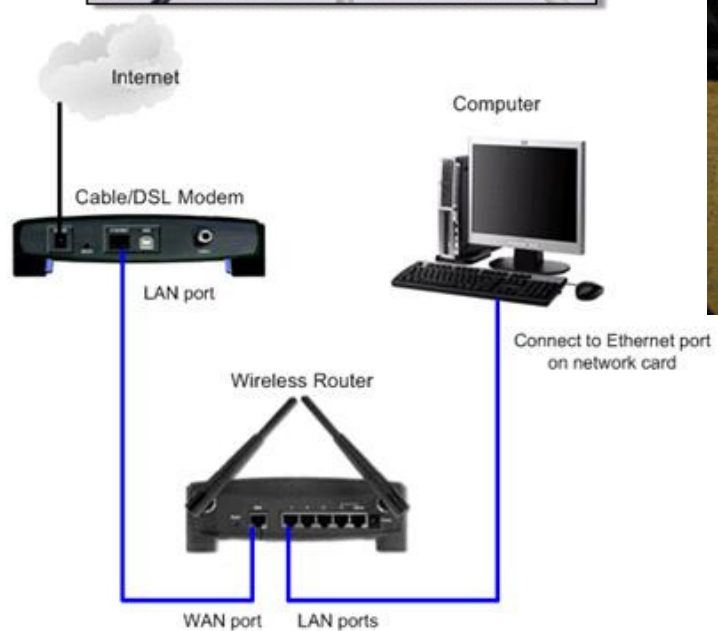
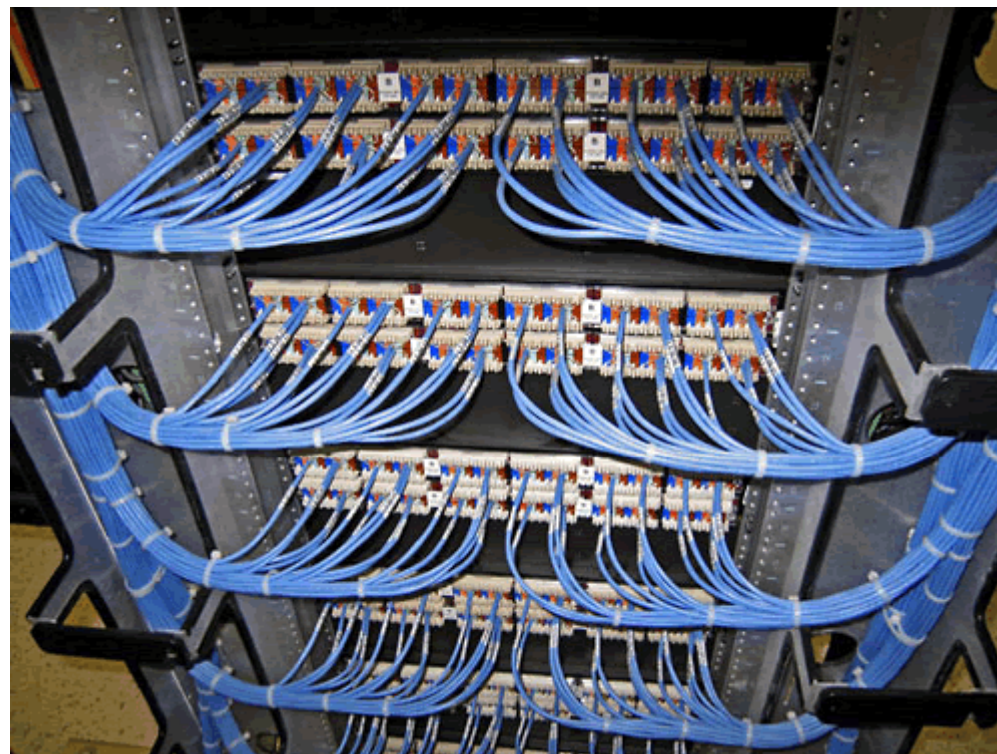
- Sissejuhatus
- Kohtvõrk: ethernet
- Internet seob kohtvõrke
- Interneti baasmehhanismi, IP funktsioneerimine
- TCP ja UDP: kaks täiendavat protokollid IP-le
- Üldpilt kihilisest protokollindusest: ISO/OSI mudel
- Muud sideprotokollid
- HTTP protokoll
- DNS funktsioneerimine
- Näiteid brauseriga ja telnetiga

## Võrkude võrk



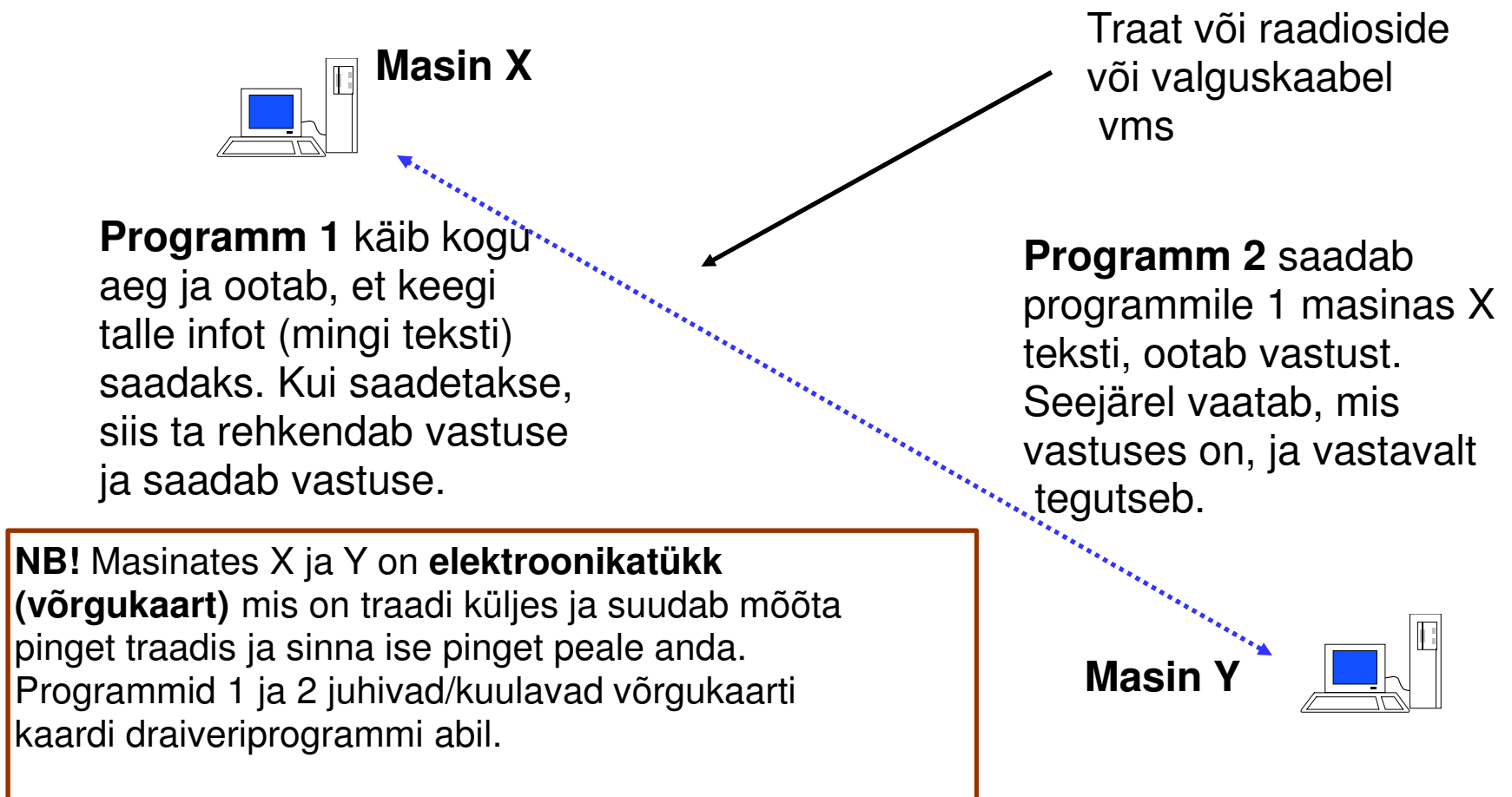


# Juhtmed, võrguseadmed, arvutid



# Kiirülevaade võrgunduse funktsioneerimisest

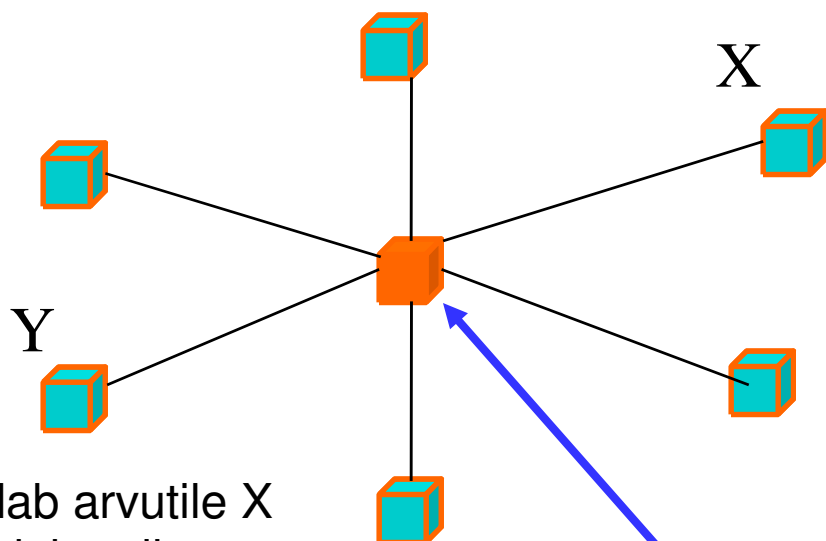
## ■ Üldpilt:



# Interneti funktsioneerimine 1

- Üldpilt sama, mis eelmisel slaidil.
- Küsimused, mis tahavad vastust:
  - Kui arvuti Y on traadiga ühenduses mitme eri arvutiga (X, Z, U, ...) siis kuidas saata nimelt soovitud arvutisse (meie näites X) oma info?
  - Kui X ja Y pole otse ühenduses vaid teiste kaudu, siis kuidas jõuavad sõnumid ühest arvutist teise?
  - Kui info hulk on väga suur (näiteks 700 MB), kas saaks seda üksikute väiksemate tükkidena saata?
  - Kuidas kontrollida, kas tükid jõudsid pärale?
  - Mis teha, kui mõni tükk kaduma läheb?
  - Kuidas arvuti X tükid õieti kokku oskab panna?

## Hulk arvuteid võrgus traatipidi koos: ethernet



Kui arvuti Y saadab arvutile X teksti, siis ta saadab selle Alguses traatipidi **KÕIGILE** arvutitele võrgus, aga teksti juures on öeldud, et **AINULT** X (MAC aadress!) peab seda kuulama. **Teised ignoreerivad.**

**Switch** on filter, mis õpib võrguliiklust kuulates millise traadi otsas on mingi masin ja saadab edasi AINULT sellele, kellele teade kuulub.

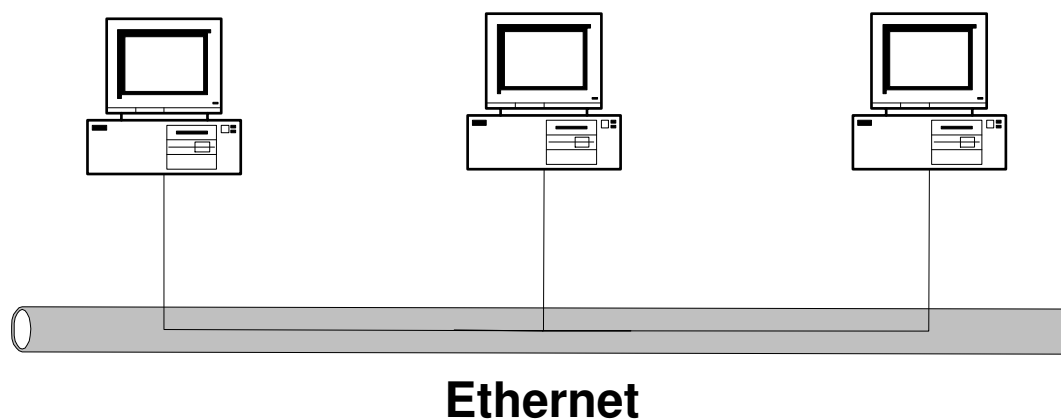
Igal arvutil on oma **unikaalne NIMI** ehk aadress **Ethernetis**:

MAC aadress:  
48 bitine arv  
01:23:45:67:89:ab

**NB! Korraga on traadil ainult üks teade!**

## Etherneti algne siini arhitektuur

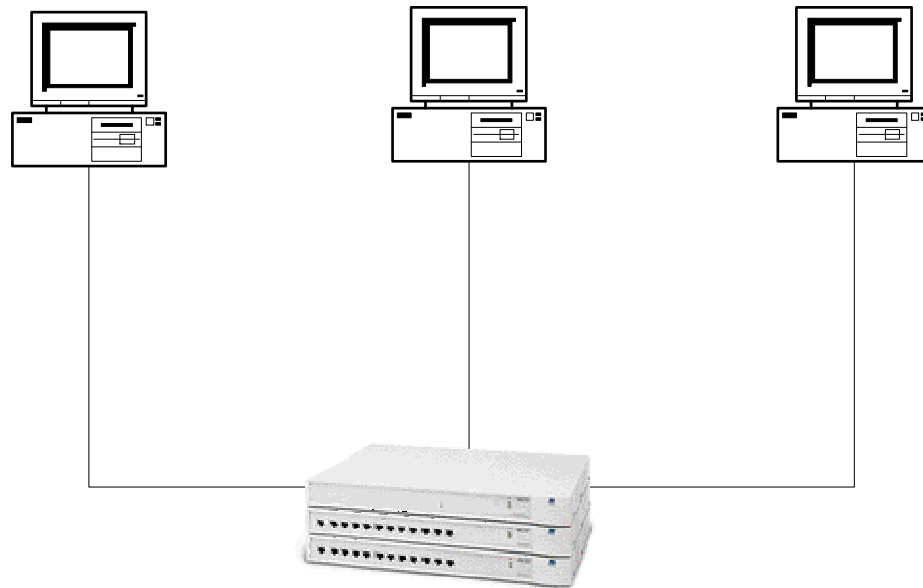
- 10Base5 ja 10Base2 Ethernet kasutas ühtset jagatud siini (eetrit)



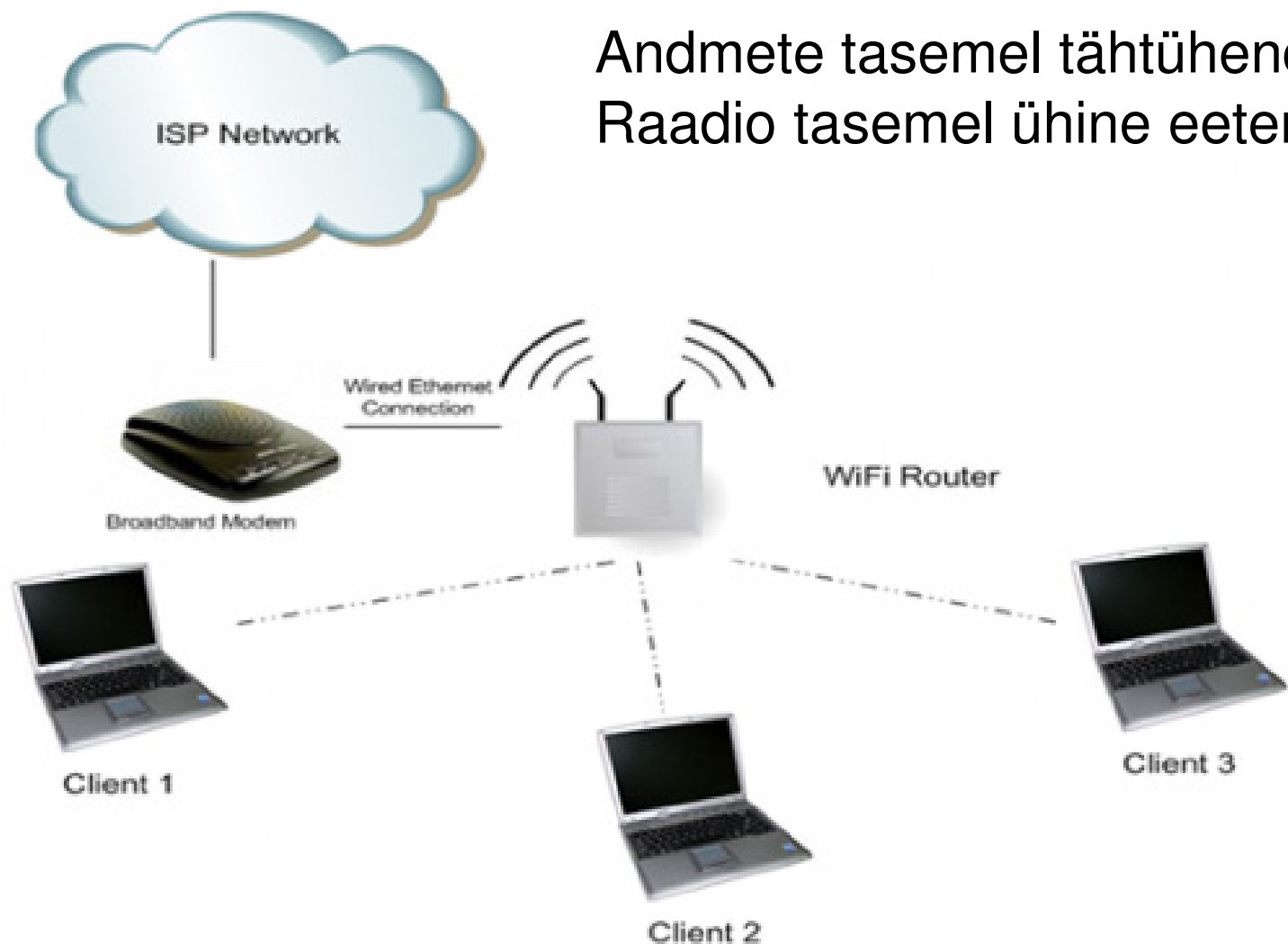


## Tänapäeval kasutatakse täht-arhitektuuri

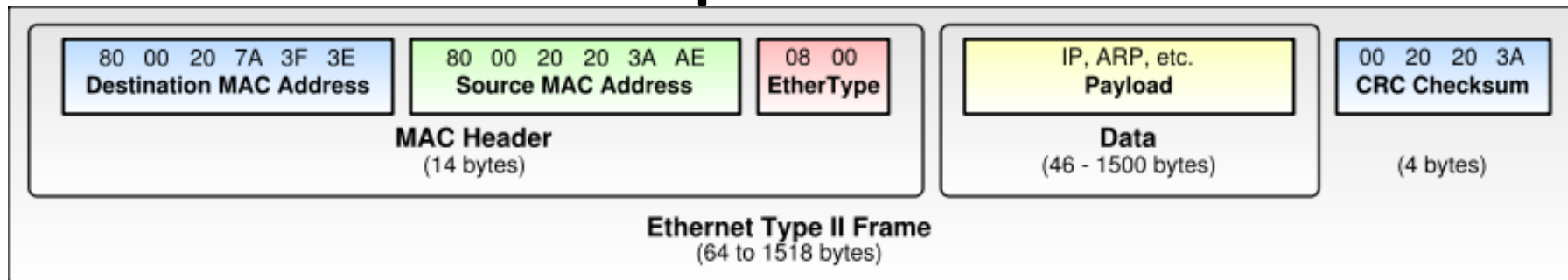
- Seadmed on ühendatud *switch*-iga



## Traadita võrk – kombinatsioon



# Ethernet paketi formaat



- Preamble 7 octets of 10101010
- Start-of-Frame-Delimiter 1 octet of 10101011
- **MAC destination 6 octets**
- **MAC source 6 octets**
- **Ethertype/Length 2 octets**
- **Payload 46-1500 octets**
- **CRC32 4 octets**
- (Postamble 1 octet of 01111110)
- Interframe gap 960 ns (100M)

## Internet?

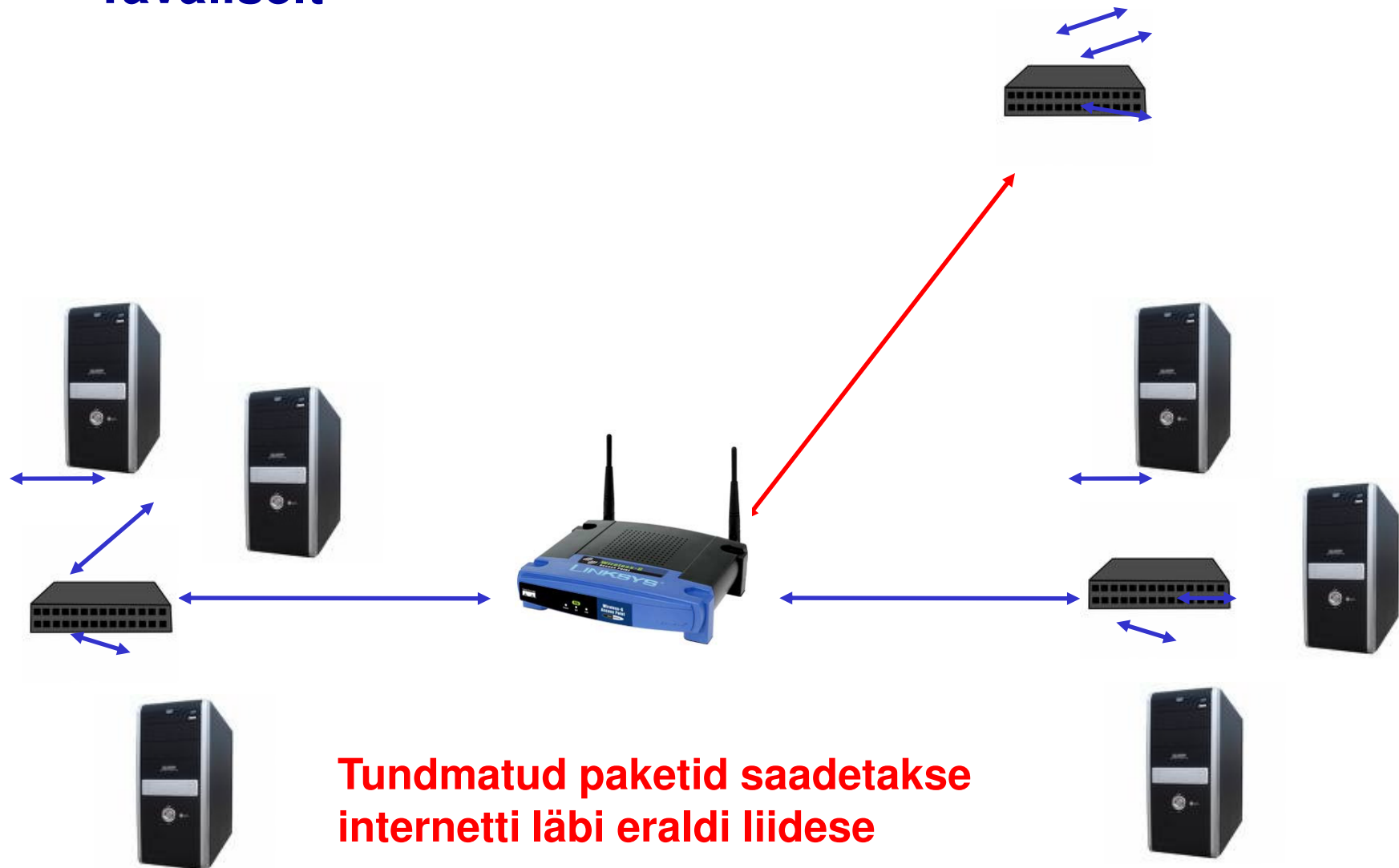
- Internet seob erinevaid kohtvõrke.
- Kohtvõrgus reeglina ethernet (sh wifi).
- Eri kohtvõrkude sidumiseks antakse igale masinale unikaalne nn IP aadress: tüüpiliselt neli baiti (193.40.254.28).
- IP aadress ja muu vajalik info on etherneti paketi sisu üks osa.
- Ruuter on masin, mis ühendatud korraga mitme kohtvõrguga ja mis otsustab, kuhu kohtvõrku pakett edasi saata.

# Marsruuter

- Saab teha suvalisest arvutist



## Tavaliselt





## Suured infrastruktuuri marsruuterid

Väga kiired, oskavad marsruutimise keerulisemaid osi



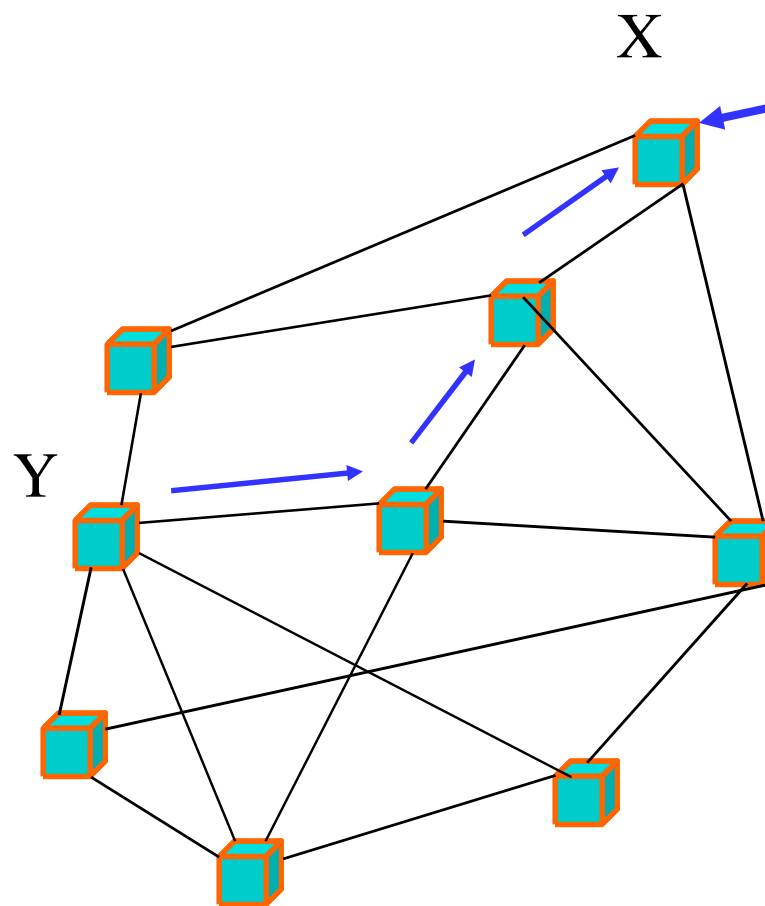
# Hulk arvuteid võrgus traatipidi koos: internet

**Teksti küljes on alati IP aadress.**

Iga võrgus olev masin suunab saadud teksti õiges suunas. Kuidas?

Nn “routing table” ütleb, vaadates IP aadressi.

Igas arvutis on nn “**router**”-programm, mis kuulab traati ja saadab saadud tekstid õiges suunas edasi.



Igal arvutil on oma **unikaalne NIMI** ehk aadress

**Internetis:**  
IP aadress:  
32 bitine arv

32 bitti on 4 baiti.  
IP aadress kirjutatakse enamasti “inimloetavalt” nii (näide):  
193.40.254.179

Osa arvuteid võrgus on “lihtsalt” harilikud arvutid, osa spetsiaalsed “võrgu-ruuterid”

# Interneti alusprotokoll (IP) kasutamise põhi-ideed

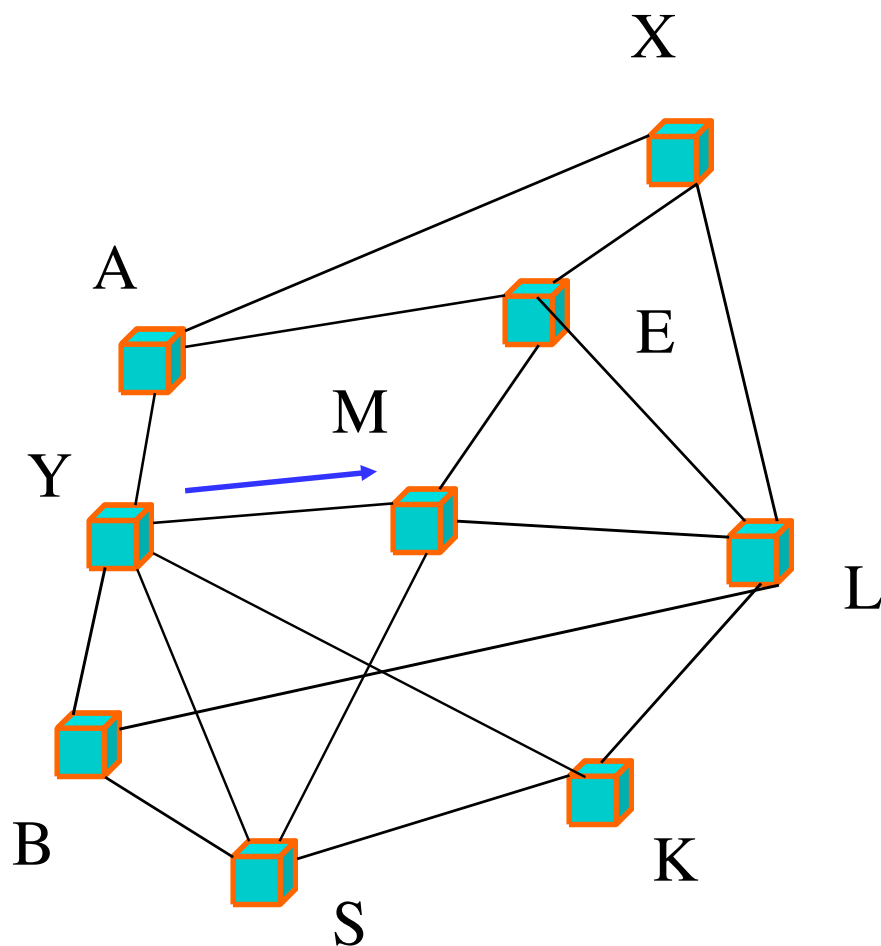
- **Aadress:** iga saadetava tekstijupiga pannakse kaasa selle masina aadress, kuhu tekst saata tuleb.
- See on analoogiline hariliku ümbrikuga:
  - tekst on ümbriku sees.
  - ümbriku peal on saaja aadress (ja tagaküljel ka saatja aadress)
  - 4 baiti IP aadressis on nagu aadressi-read: maja, tänav, linn, riik
  - ruuter-arvutid on nagu postkontorid ja sorteerimispunktid
  - traadid on nagu veomasinad, mis posti kontorite vahel veavad
- **Paketid:** harilikku ümbrikku aga ei mahu terve entsoklüpeedia kümme köidet! Mida teha? Paneme eraldi köited eri ümbrikutesse ja saadame kümme ümbrikku.
- Nii ka internetis:
  - Pikad tekstid lõhutakse väiksemateks juppideks (nn pakettideks)
  - Paketid saadetakse eraldi, igaühel aadress peal.

## Kuidas info siis liigub? 1

Y tahab saata teksti  
“Ahoi, tere”  
masinale IP-  
aadressiga X.

Ruuterprogramm  
otsustab, kumbat  
juhet (tegelikult,  
vahemasina  
aadressi) valida.

Otsutagu näiteks  
masina M kasuks.

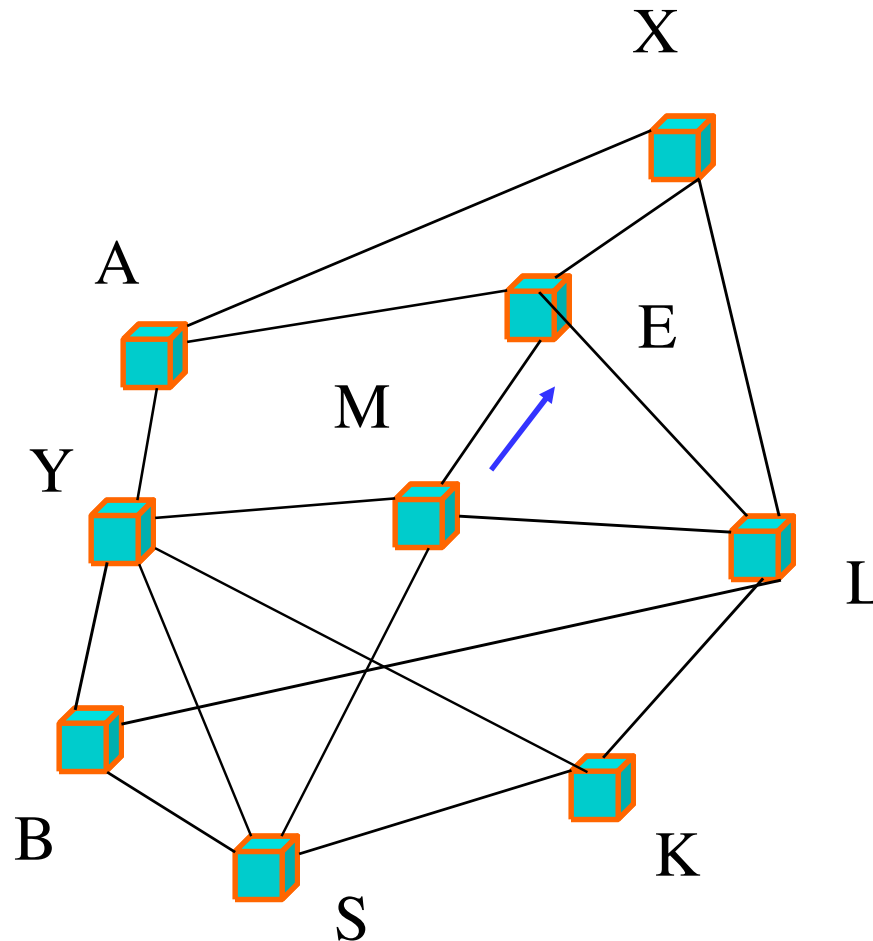


## Kuidas info siis liigub? 2

M peab saatma teksti “Ahoi, tere” edasi masinale IP-aadressiga X.

Ruuterprogramm otsustab, kumbat juhet (tegelikult, vahemasina aadressi) valida.

Otsutagu näiteks masina E kasuks.

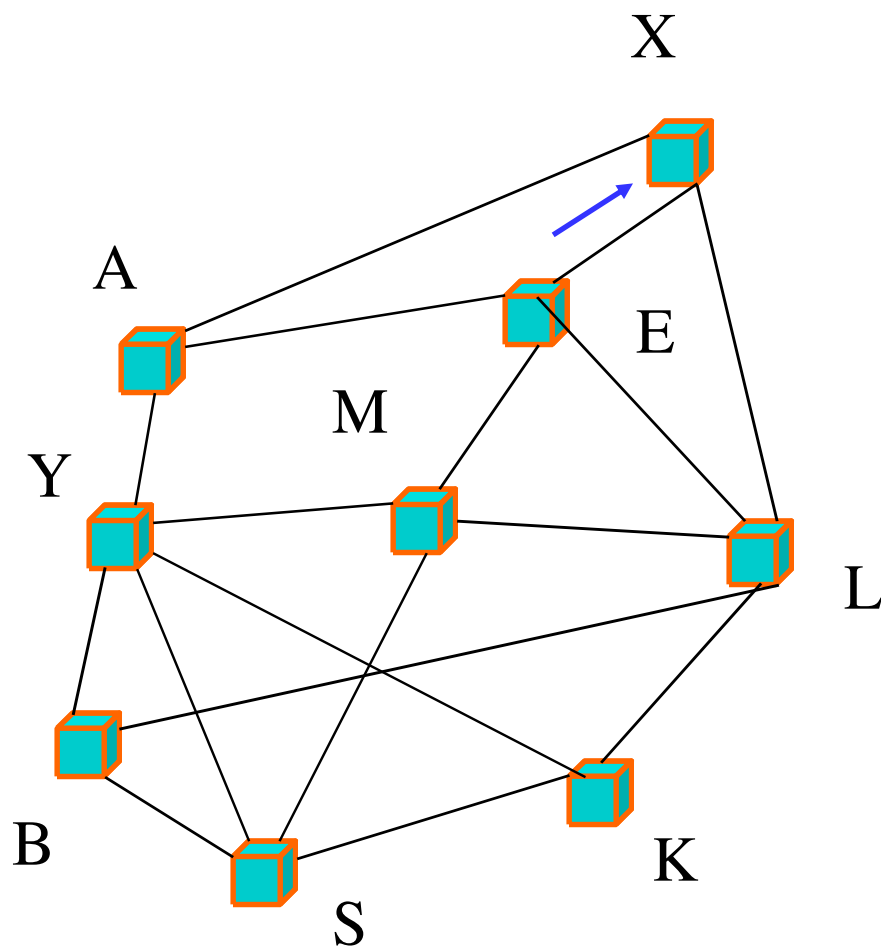


## Kuidas info siis liigub? 3

E peab saatma teksti “Ahoi, tere” edasi masinale IP-aadressiga X.

Ruuterprogramm otsustab, kumbat juhet (tegelikult, vahemasina aadressi) valida.

Otsutagu (ilmselt!) masina X kasuks.



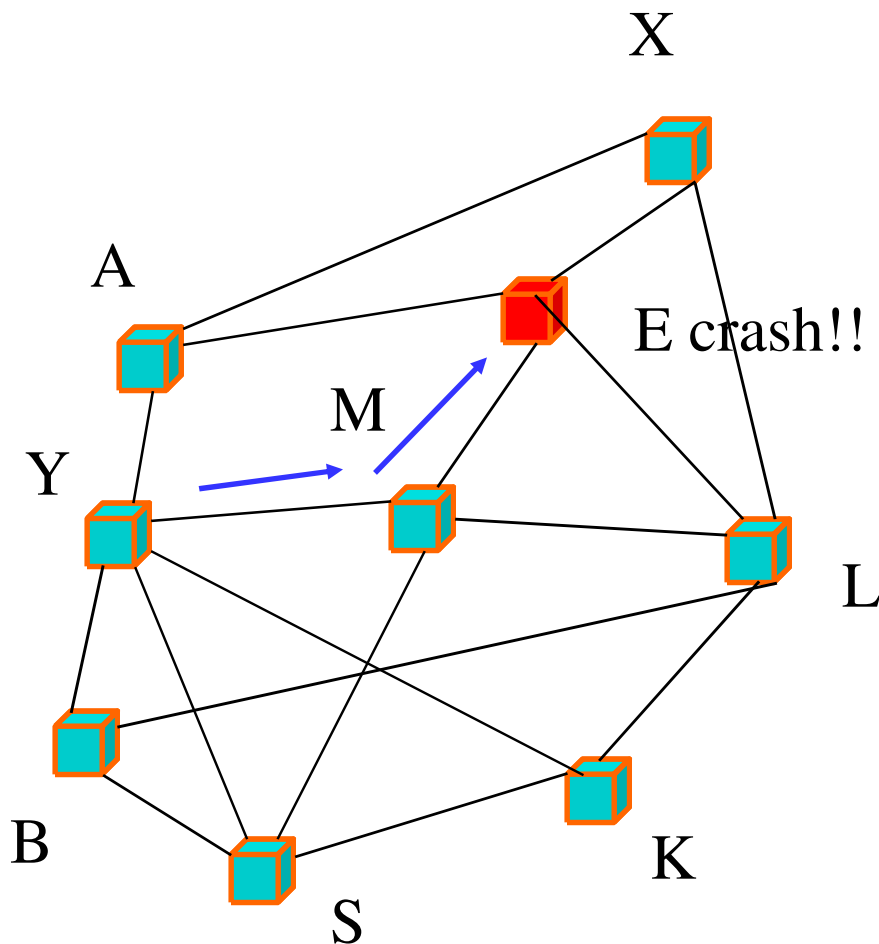


## Paketid võivad kaduma minna

Y peab saatma teksti “Ahoi, tere” edasi masinale IP-aadressiga X.

Teade läheb masinani E, aga siis E crashib (läheb katki, programm läheb segamini, vool läheb ära vms).

**Võrgu alusprotokoll: IP ise ei kontrolli, kas teade jõudis pärale!**



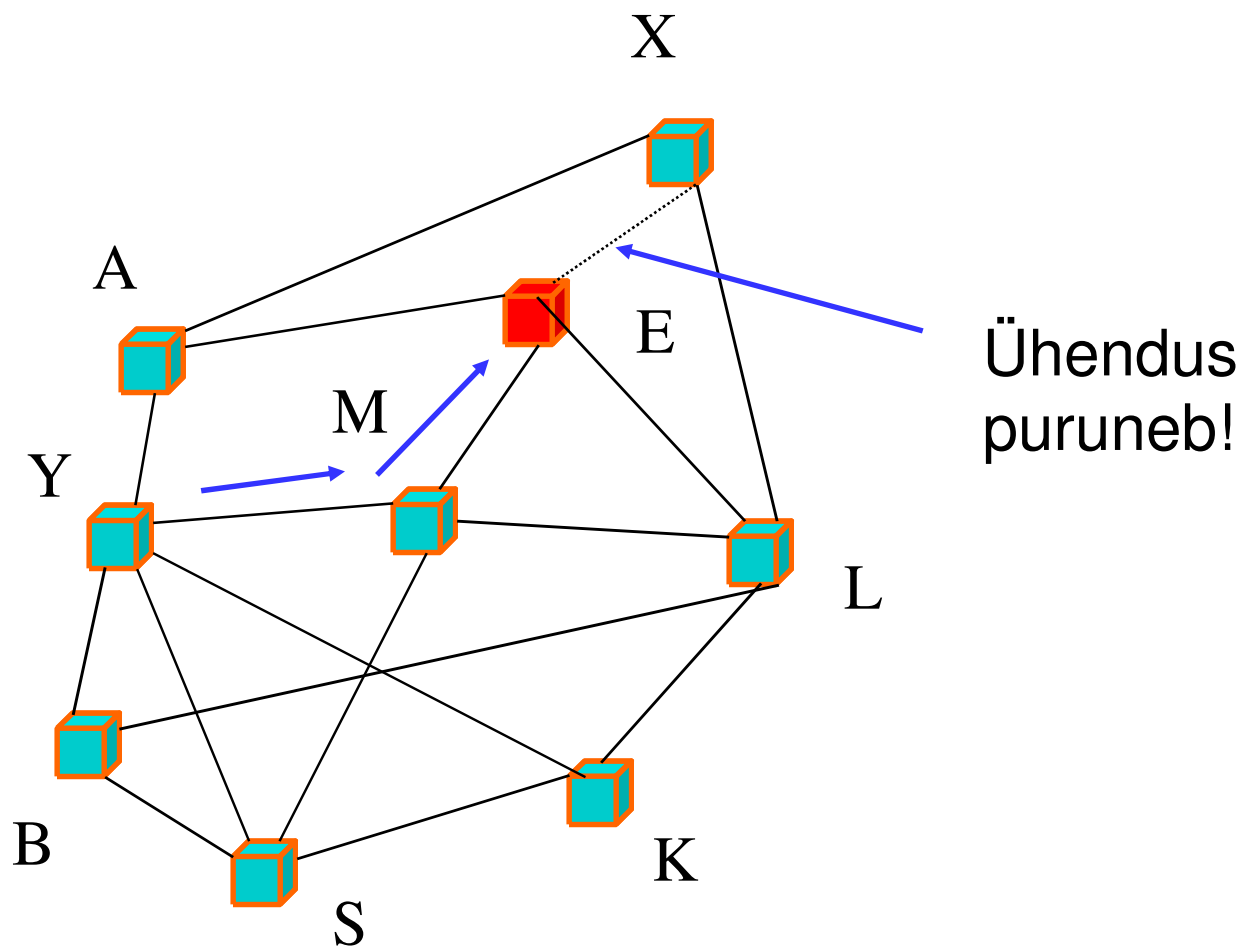
## Paketid võivad minna eri teid pidi, eri kiirusega

Y peab saatma teksti "**Ahoi, tere**" edasi masinale IP-aadressiga X.

A lõhub pika teksti kaheks paketiks!

Osa "**Ahoi**," läheb otse E->X, aga siis läheb E-X otseühendus rikki.

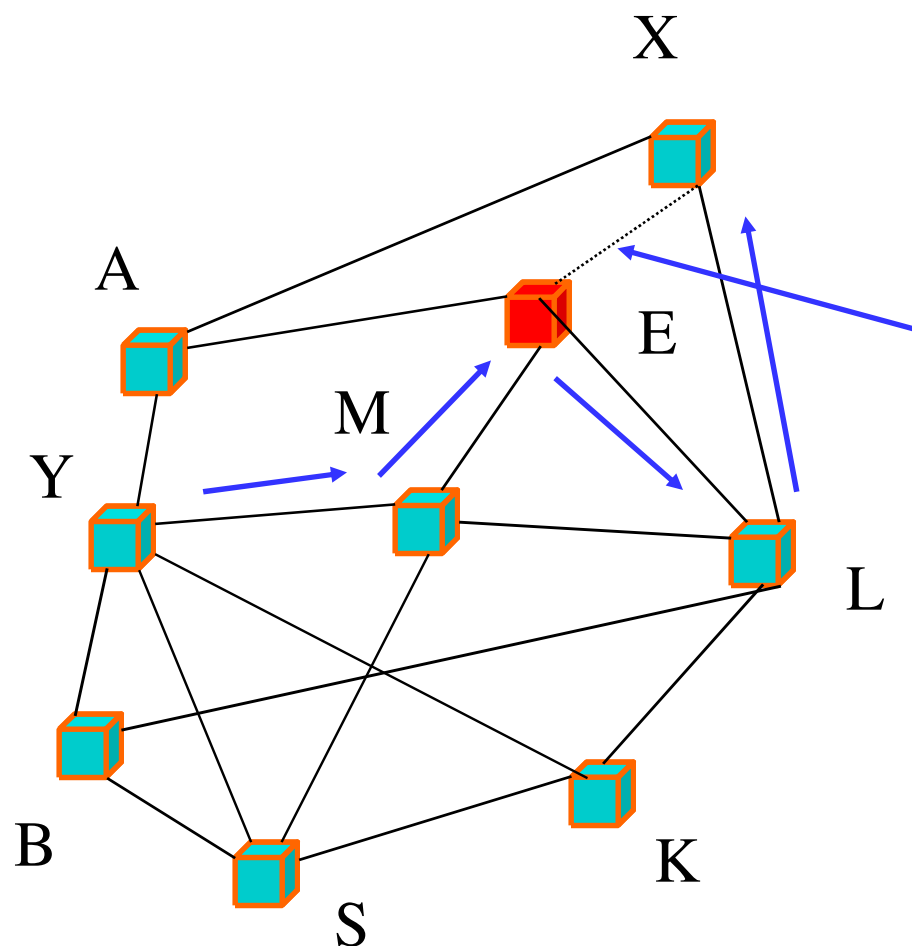
E leiab, et saab saata teate teise osa "**tere**" X-ile ka masina L kaudu.



## Paketid võivad minna eri teid pidi, eri kiirusega

E suunabki teate masinale L, kes suunab selle masinasse X.

**Kaks teksti osa liikusid eri teid pidi!**

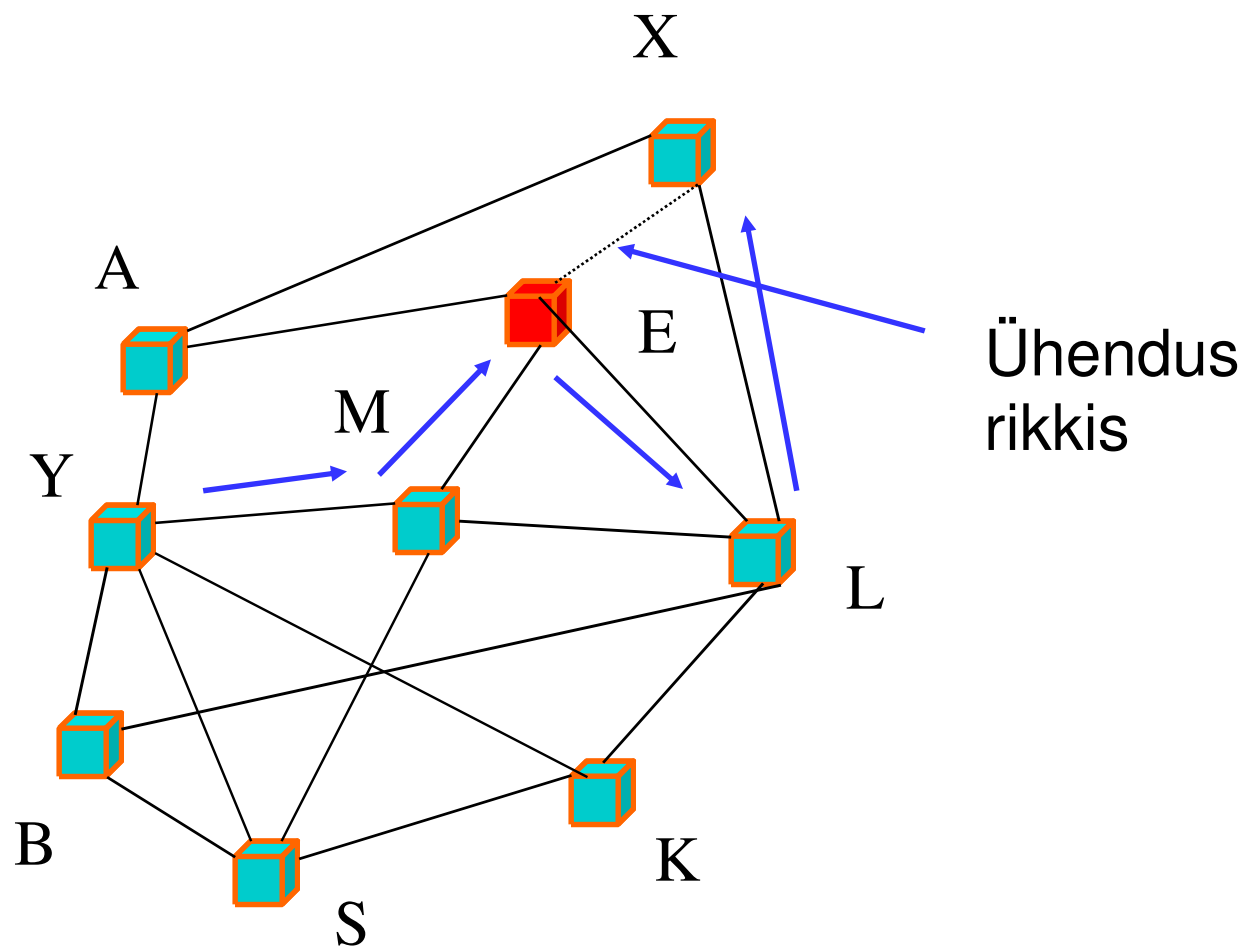


Ühendus purunenud, aga pole hullu ...

## Paketid võivad minna eri teid pidi, eri kiirusega

Hiljem saadab  
Y uue teate "**Tere  
taas**" masinale X.

E suunab esimese  
osa "**Tere**" läbi  
masina L.

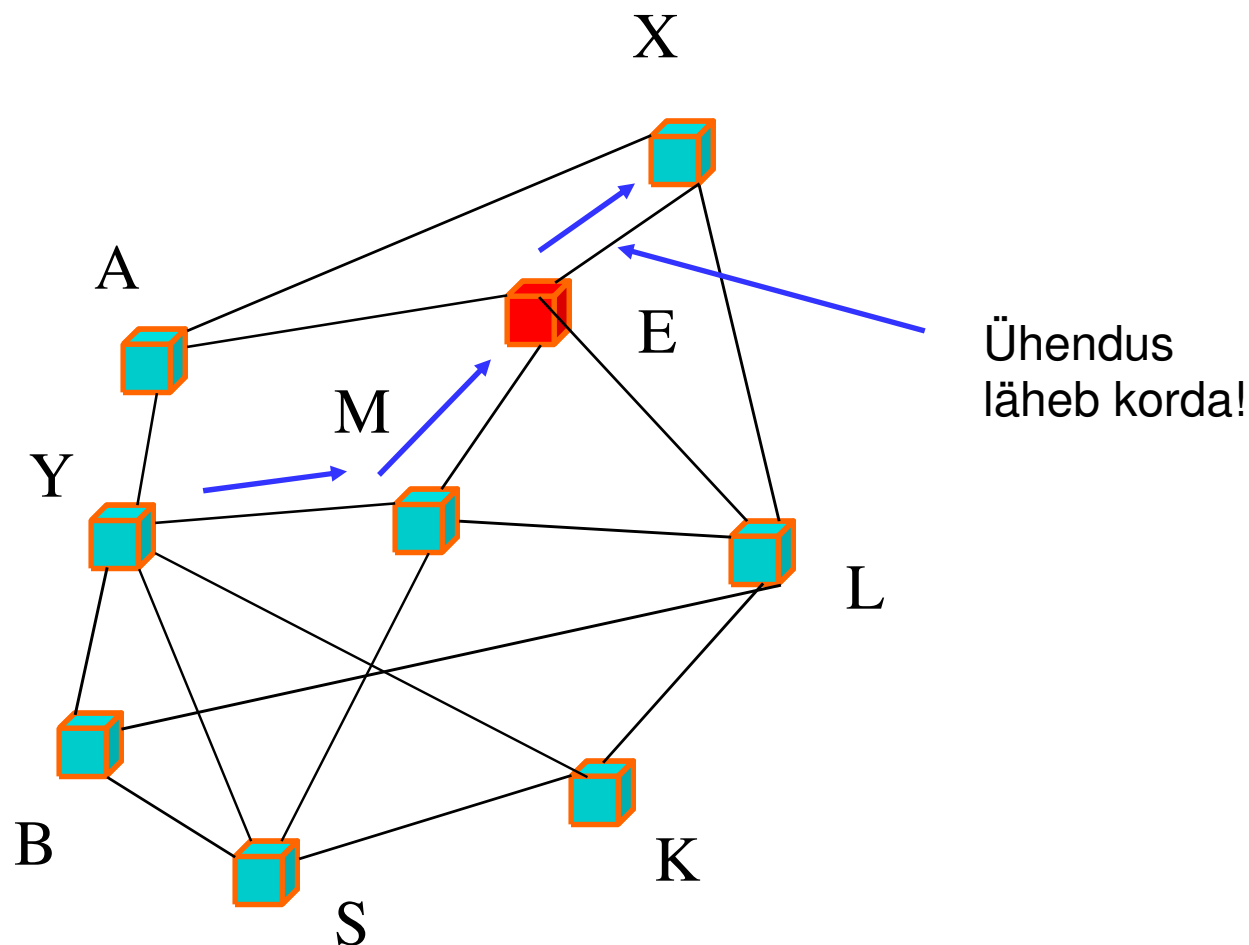


## Paketid võivad minna eri teid pidi, eri kiirusega

....

Siis läheb  
otseühendus korda,  
ja E suunab osa  
“**taas**” otse  
masinale X.

**Teine osa võib  
seega jõuda  
kohale enne  
esimest!**



## Interneti alusprotokoll: IP (internet protocol)

- IP protokoll on **kokkulepe**, et kuidas infot saata ja sellest aru saada tuleb.
- IP protokoll lubab saata ainult suhteliselt väikeseid sõnumeid (64 kB).
- Iga sõnumiette pannakse lisainfo (päis ehk header), mis ütleb, et:
  - kuhu see sõnum saata tuleb (IP aadress)
  - kust sõnum tuli (saatja IP aadress)
  - Hulga lisainfot ka veel



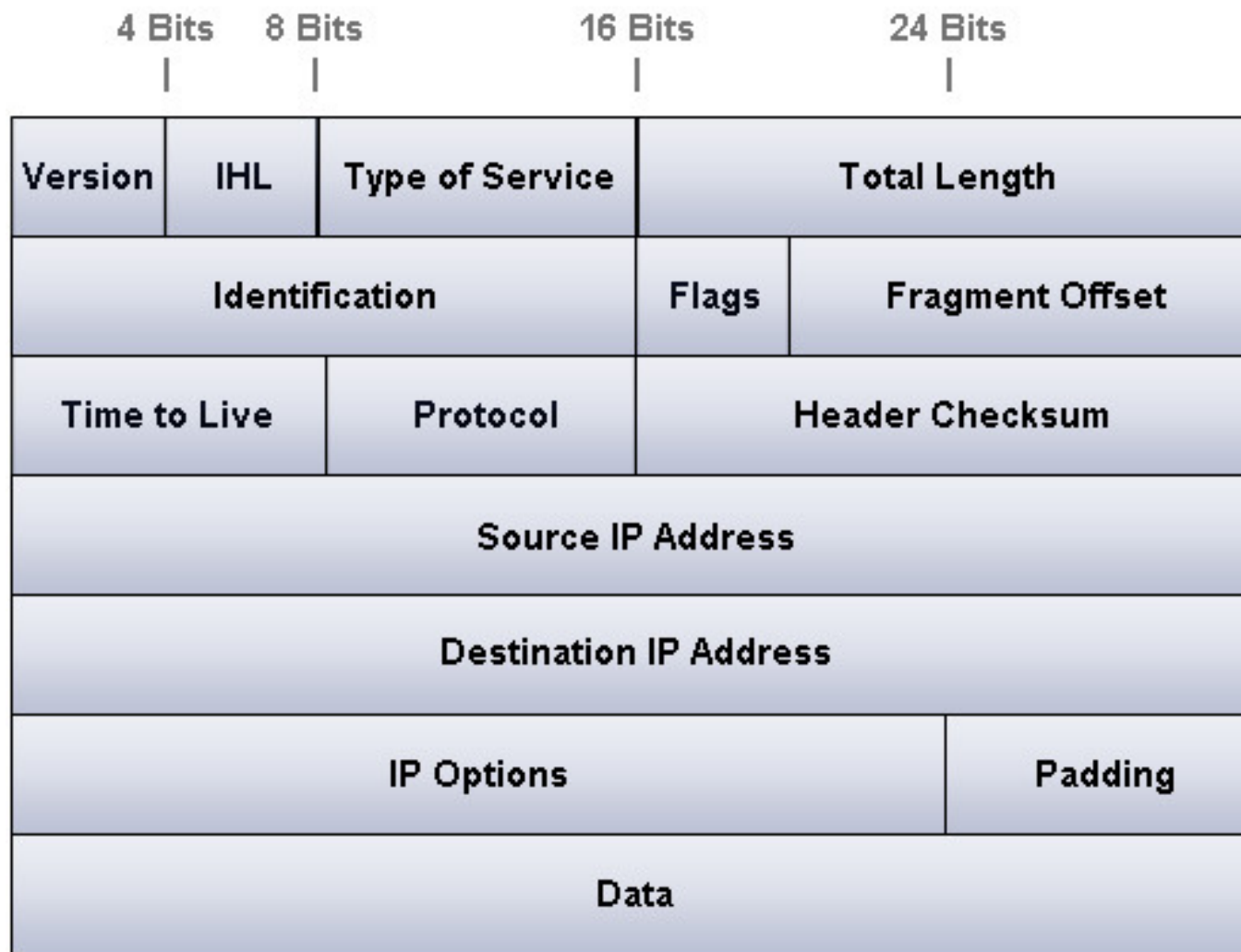


## IP (Internet Protocol)

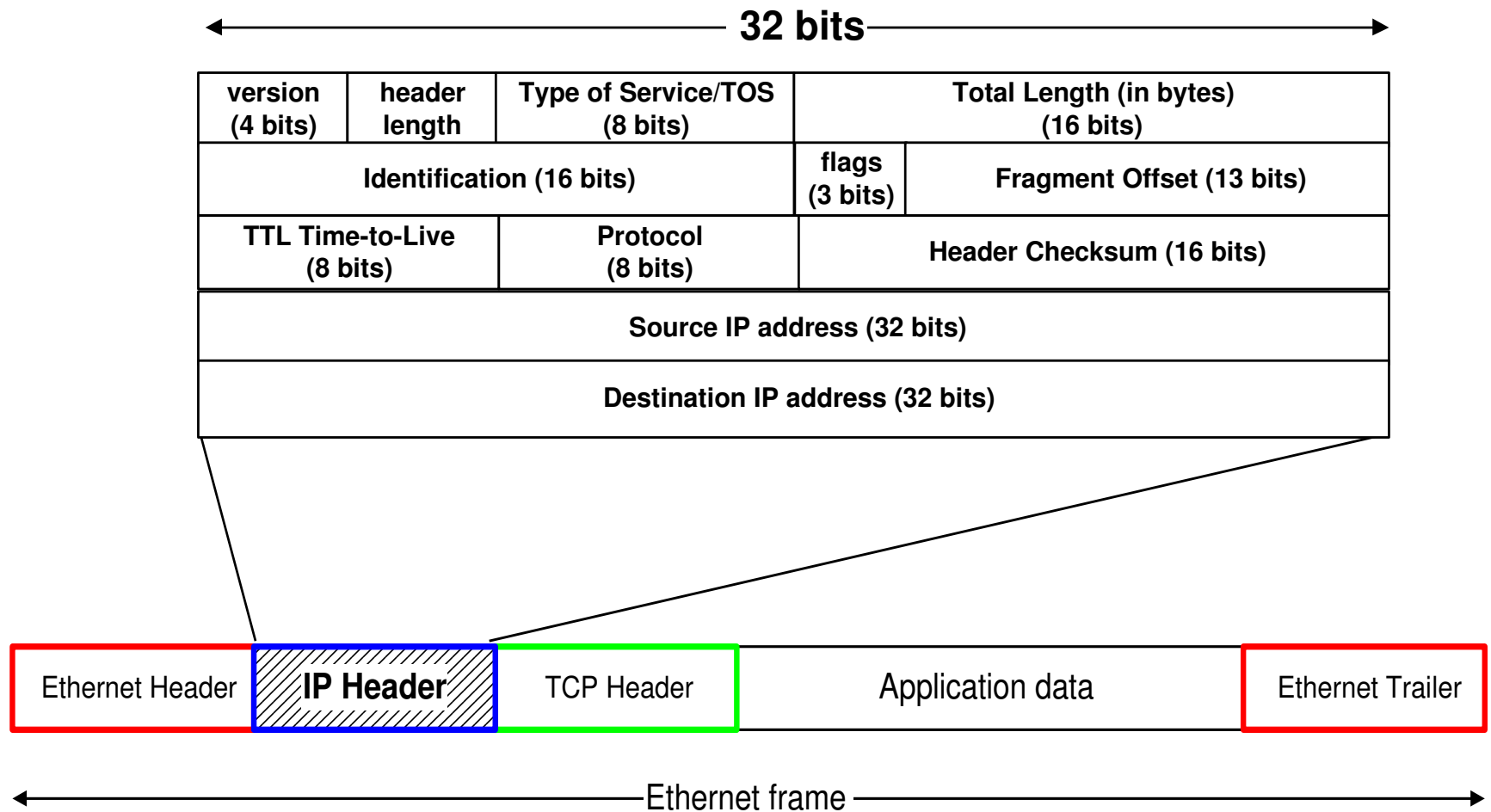
- TCP/IP pere tööhobune
- RFC 791
- Garanteerib marsruutimise, st minemise õiges suunas
- Mitteusaldusväärne - ei taga kohalejõudmist
  - Kui sisendpuhver on täis, siis ignoreerib
- Ei loo kanalit
  - iga datagrammi käsitletakse sõltumatult

# Interneti alusprotokoll: IP (internet protocol)

- Konkreetsemalt (pole vaja pähe jätta!)

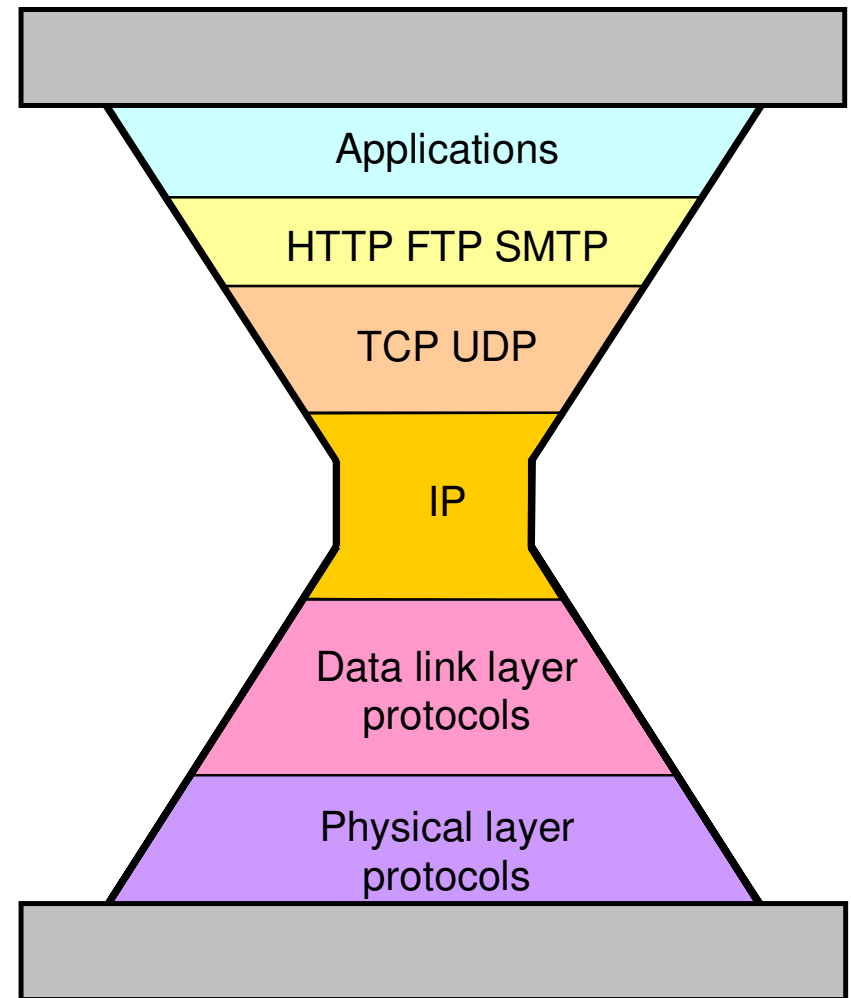


## IP pakett etherneti paketi sees



# IP: liivakella kitsas koht

- IP on Interneti protokollide liivakella taljeosa
- Mitmeid kõrgema taseme protokolle
- Mitmeid madalama taseme protokolle
- Ainult üks protokoll võrgu tasemel



# Kanali- ja paketipõhised protokollid

## Kanalipõhised protokollid

Näiteks: telefoniühendus

nimetatakse ka *olekuga (stateful)* protokollideks

osapoolle loovad (virtuaalse) kanali

enne andemvahetust on vaja kanal luua ja pärast sulgeda

osapooled teavad teise olekut kommunikatsiooniprotsessis

kommunikatsioon on “vestlus”

näiteks SMTP, TCP, ATM

## Paketipõhised protokollid

Näiteks: postiühendus

nimetatakse ka *olekuta (stateless)* protokollideks

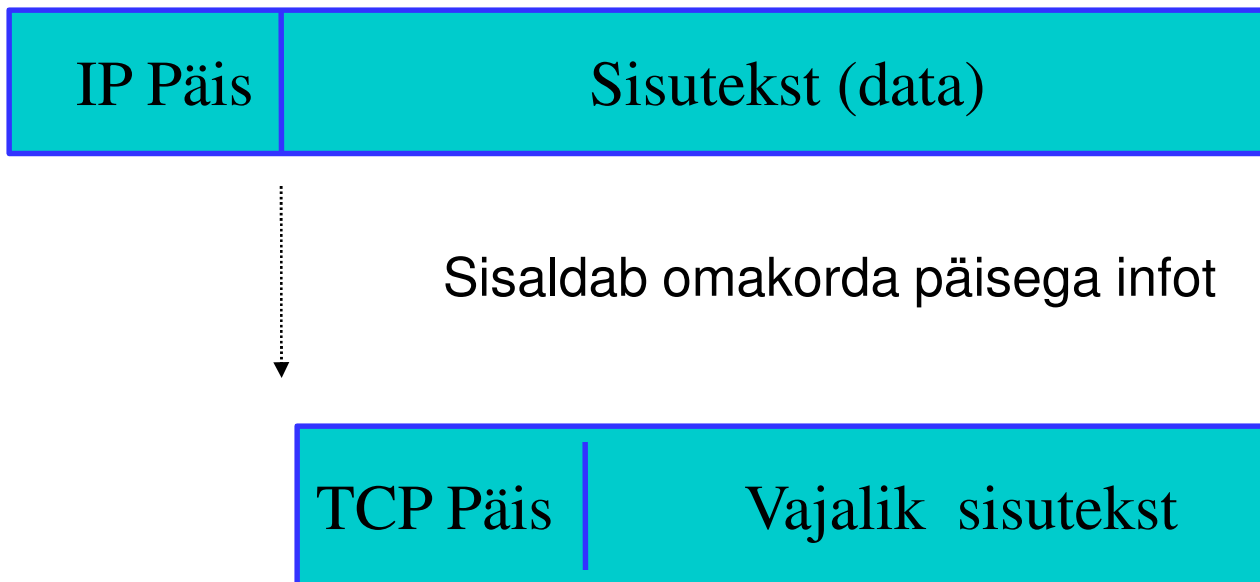
suhtlus toimub paketi kaupa

kommunikatsioon on “küsimus-vastus vormis”

näiteks DNS, NFS, UDP, IP, Ethernet

# Interneti järgmised protokollid: UDP ja TCP

- Kaks põhi-protokolli, mis kasutavad IP-d.
  - **UDP** (user datagram protocol). Ei kontrollita, kas info jõudis päralt.
  - **TCP** (transfer control protocol). Toimub kontroll.
- Kumbalgi puhul (TCP näitel):





## UDP (User Datagram Protocol)

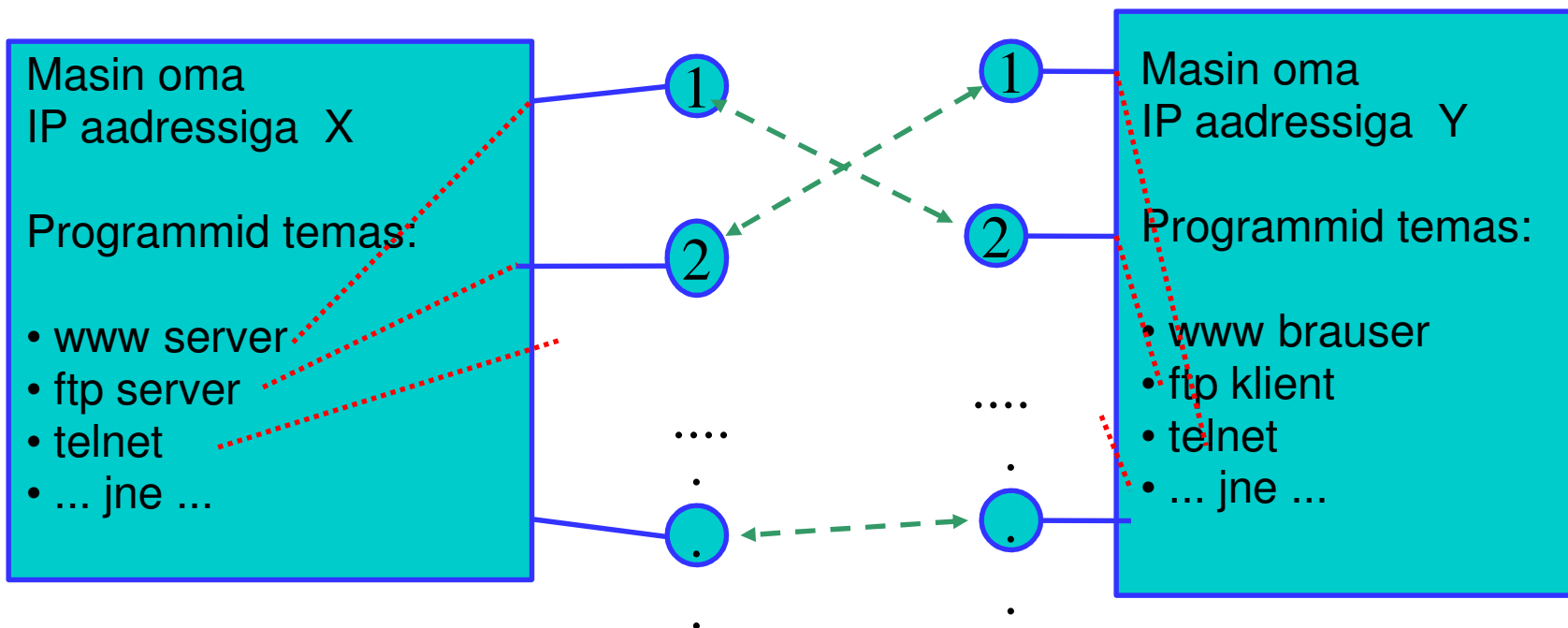
- Iga rakenduse väljund tekitab uue datagrammi
- Ei taga usaldatavust
- Datagrammi ehitus:
  - lähte- ja sihtport (kumbki kaks baiti)
  - datagrammi pikkus (kaks baiti)
  - kontrollsumma (kaks baiti, pole kohustuslik)
  - andmeosa (varieeruva pikkusega hulk baite)
- UDP paketi maksimaalpikkus on seega 64 kilobaiti.
- Kontrollsumma vea puhul unustatakse datagramm
- Rakendused: DNS, NFS, TFTP

# TCP(Transmission Control Protocol)

- Ühendusorienteeritud
- Usaldatav
- Voo tüüpi
  - Jagab voo segmentideks
  - Saates käivitab taimeri ja ootab kinnitust
  - Kinnitab saadud segmendid
  - Kontrollsumma päisest ja andmetest
  - Korrastab segmentide järjestuse
  - Unustab dublikaadid
  - Kontrollib voo mahtu

## Pordid

- Masinas elab korraga palju programme ja igaüks tahab saada/saata oma pakette. Millisele programmile konkreetne pakett saata?
- Programmidele antakse kasutamiseks **nummerdatud TCP või UDP port (need ei ole füüsilised pistikud!)**
- Ühenduse määravad: IP aadressid, pordid, protokoll



# Pordid

- Portide liigitus:

- 1 - 1023 üldtuntud pordid, sh

- 25 SMTP – TCP

- 80 HTTP – TCP

- 517 UDP

- 1024 – 5000 klientprogrammide ajutised pordid

- >5000 muud (mitte-üldtuntud) serverid

# Klient - server mudel

## ■ Üldine algoritm

- server ootab mingil üldteada pordil ühenduskutseid
- klient reserveerib dünaamilise pordi
- klient saadab serverile ühenduskutse (koos oma pordinumbriga)
- server vastab kliendile tema pordinumbril

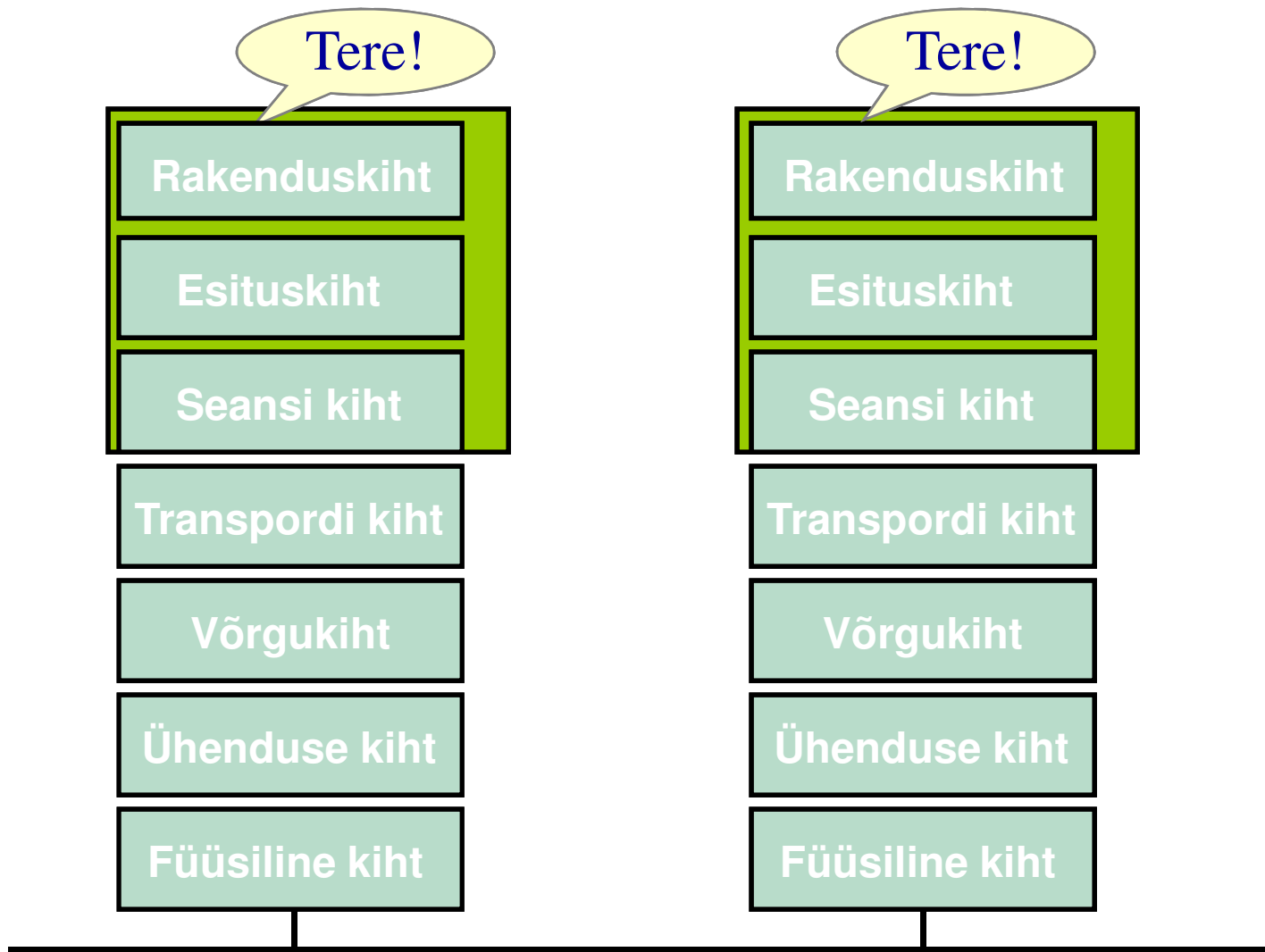
## ■ Itereeriv server

- Ei võta vastu uut kutset enne eelneva töötlemist

## ■ Paralleelne server

- Käivitab iga kutse jaoks eraldi serveri ajutise pordinumbriga selle ühenduse jaoks.
- Seega on mitme ajas lähestikuse kutse järel masinas käimas mitu “koopiat” ehk protsessi või threadi ühest serverist
- Kasutajatele paistab, et need “koopiad” käivad korraga, tegelikult hoolitseb selle “kooskäimise simulatsiooni” eest opsüsteem.

## ISO - OSI mudel



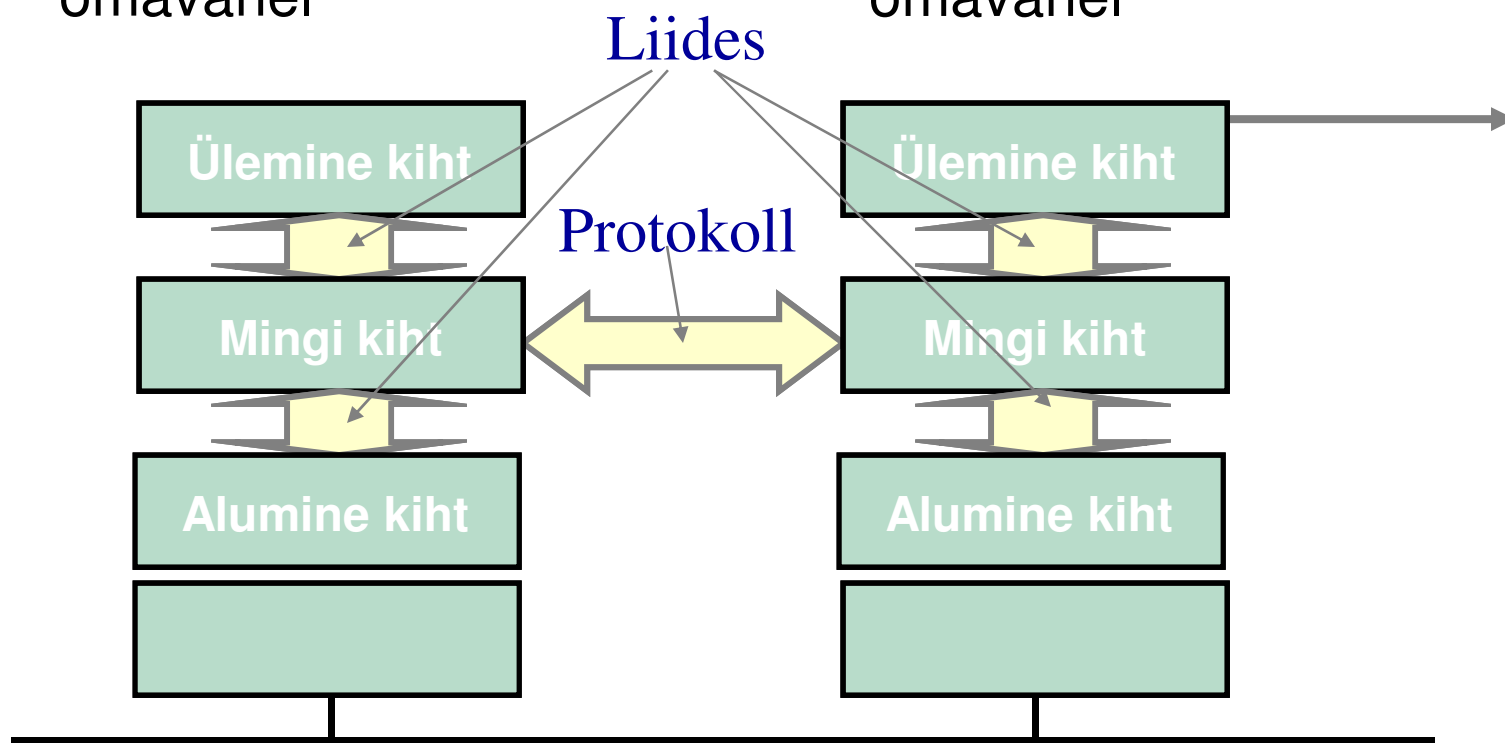
## OSI mudeli kihid

- **Protokoll (protocol)**

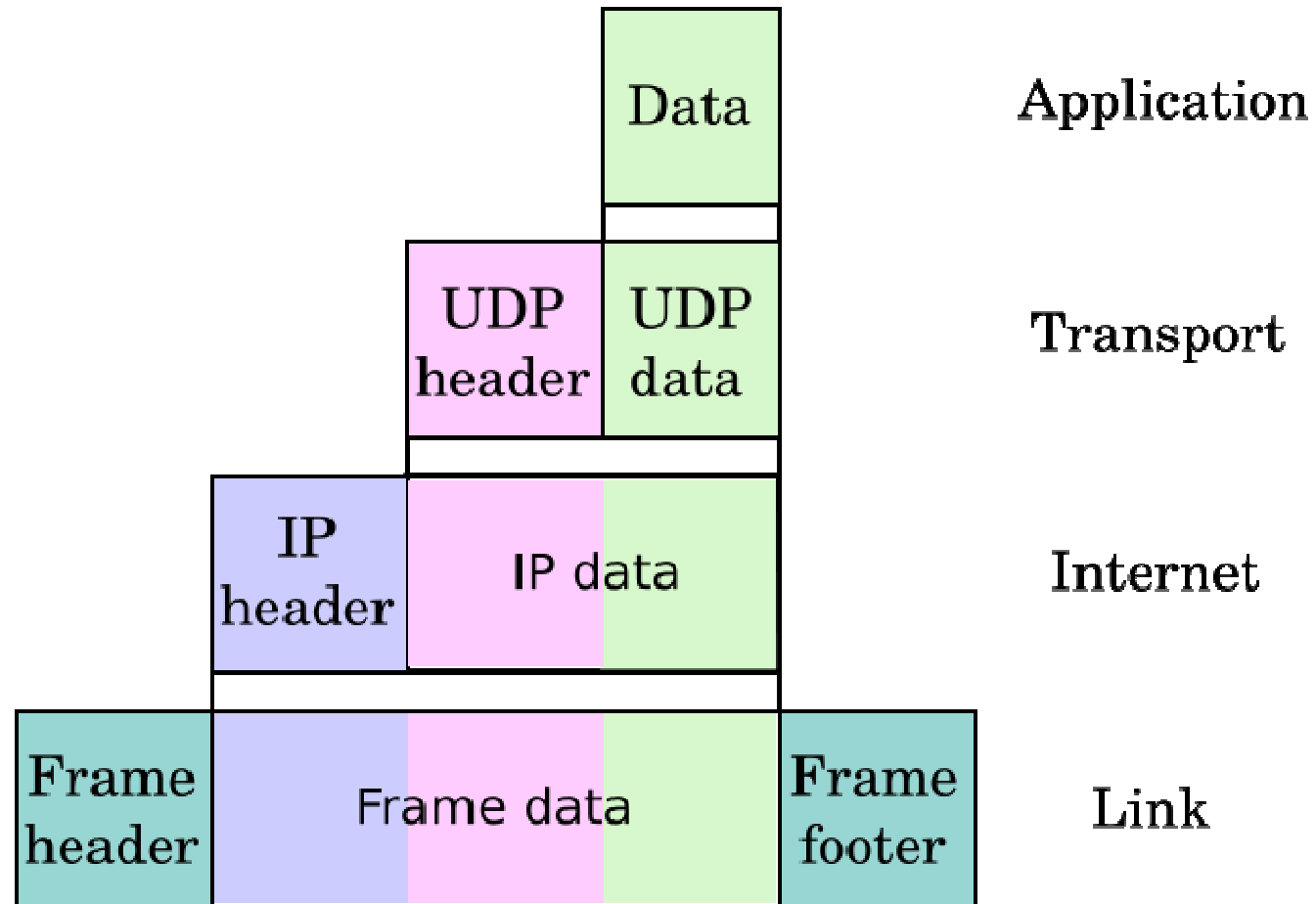
Eri süsteemide samade kihtide suhtlusviis omavahel

- **Liides (interface)**

Sama süsteemi eri kihtide suhtlusviis omavahel



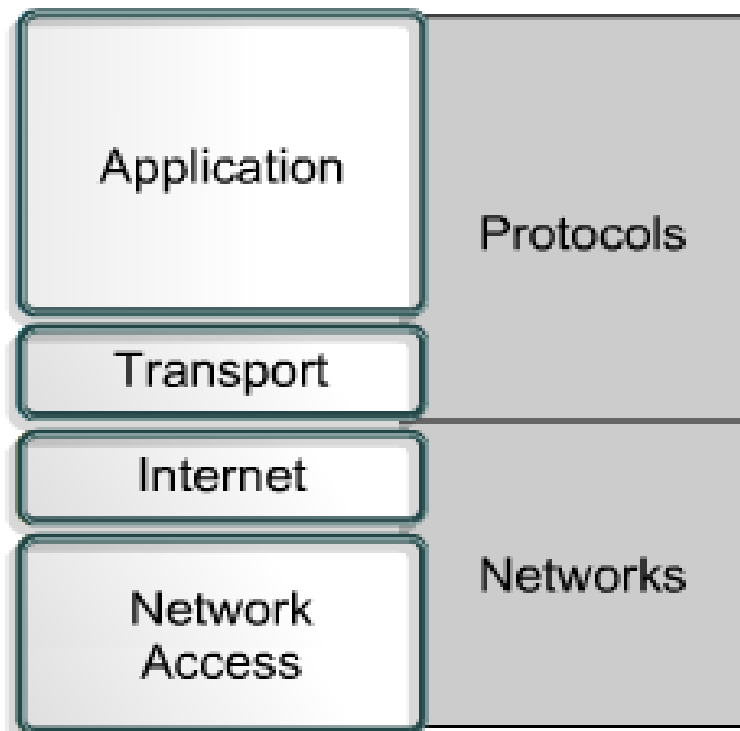
# Kapseldamine



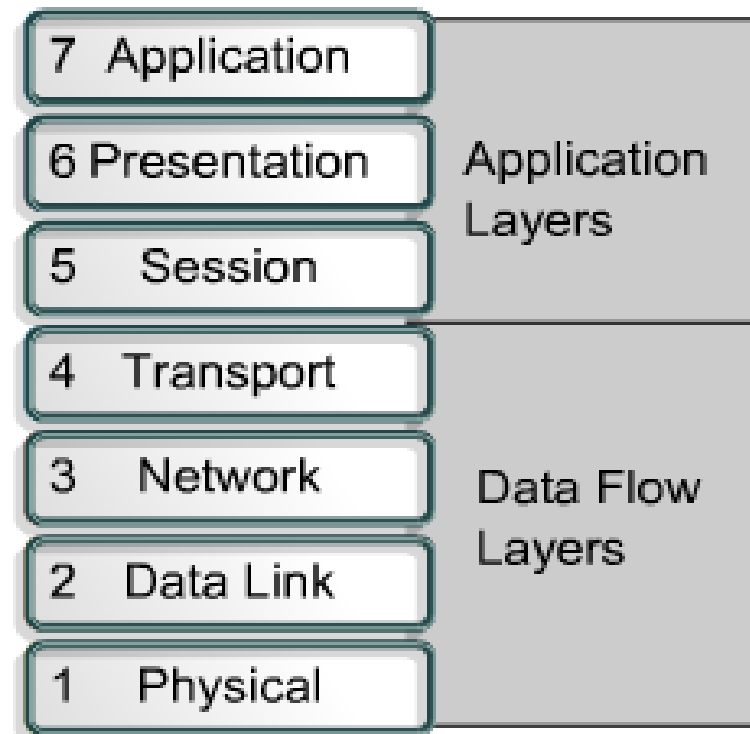


# ISO-OSI põhimõte ja internetiprotokollid: võrdlus

**TCP/IP Model**



**OSI Model**



# Kanali- ja paketipõhised protokollid

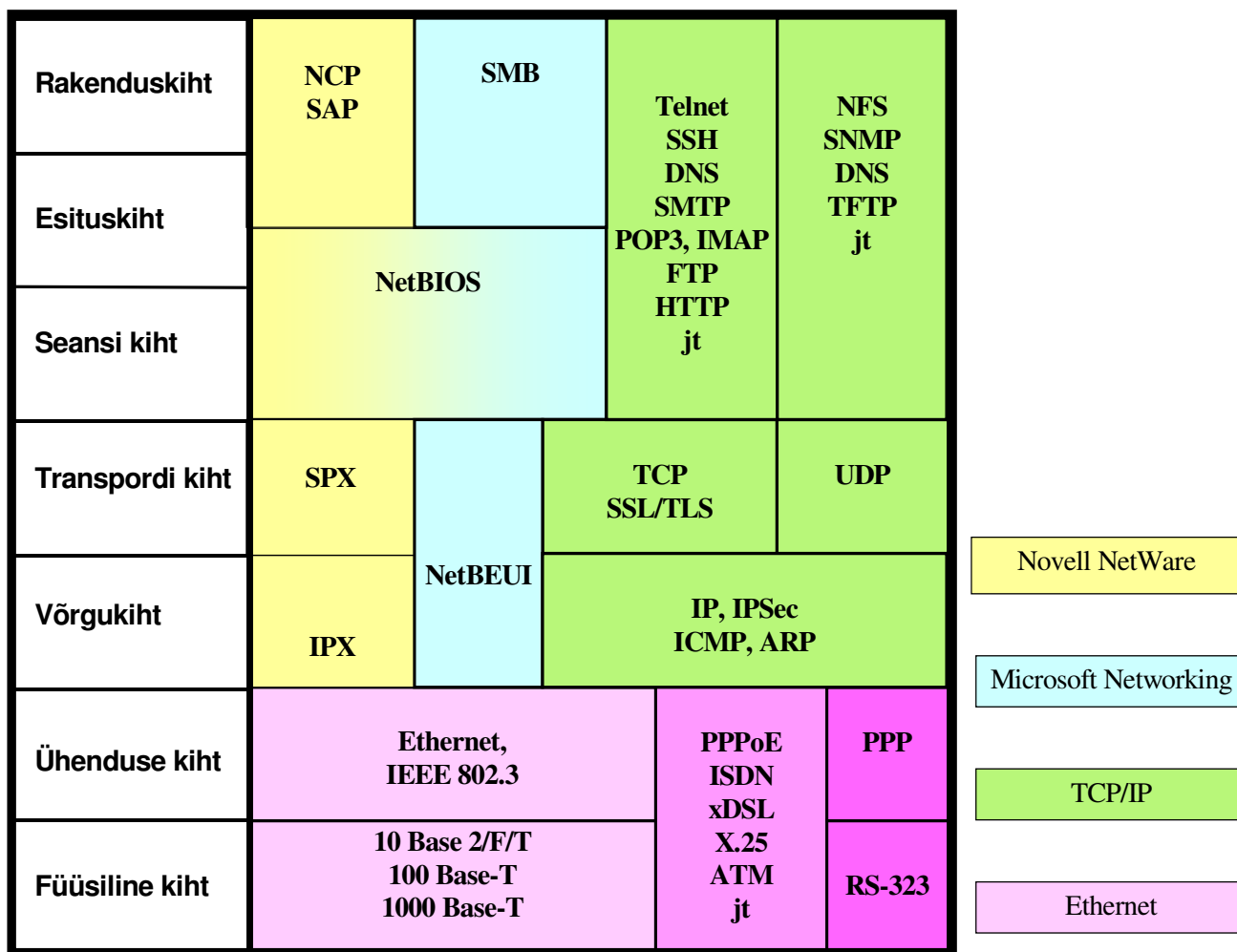
## Kanalipõhised protokollid

nimetatakse ka *olekuga (stateful)* protokollideks  
osapoolle loovad (virtuaalse) kanali  
enne andemvahetust on vaja kanal luua ja pärast sulgeda  
osapooled teavad teise olekut kommunikatsiooniprotsessis  
kommunikatsioon on “vestlus”  
näiteks SMTP, TCP, ATM

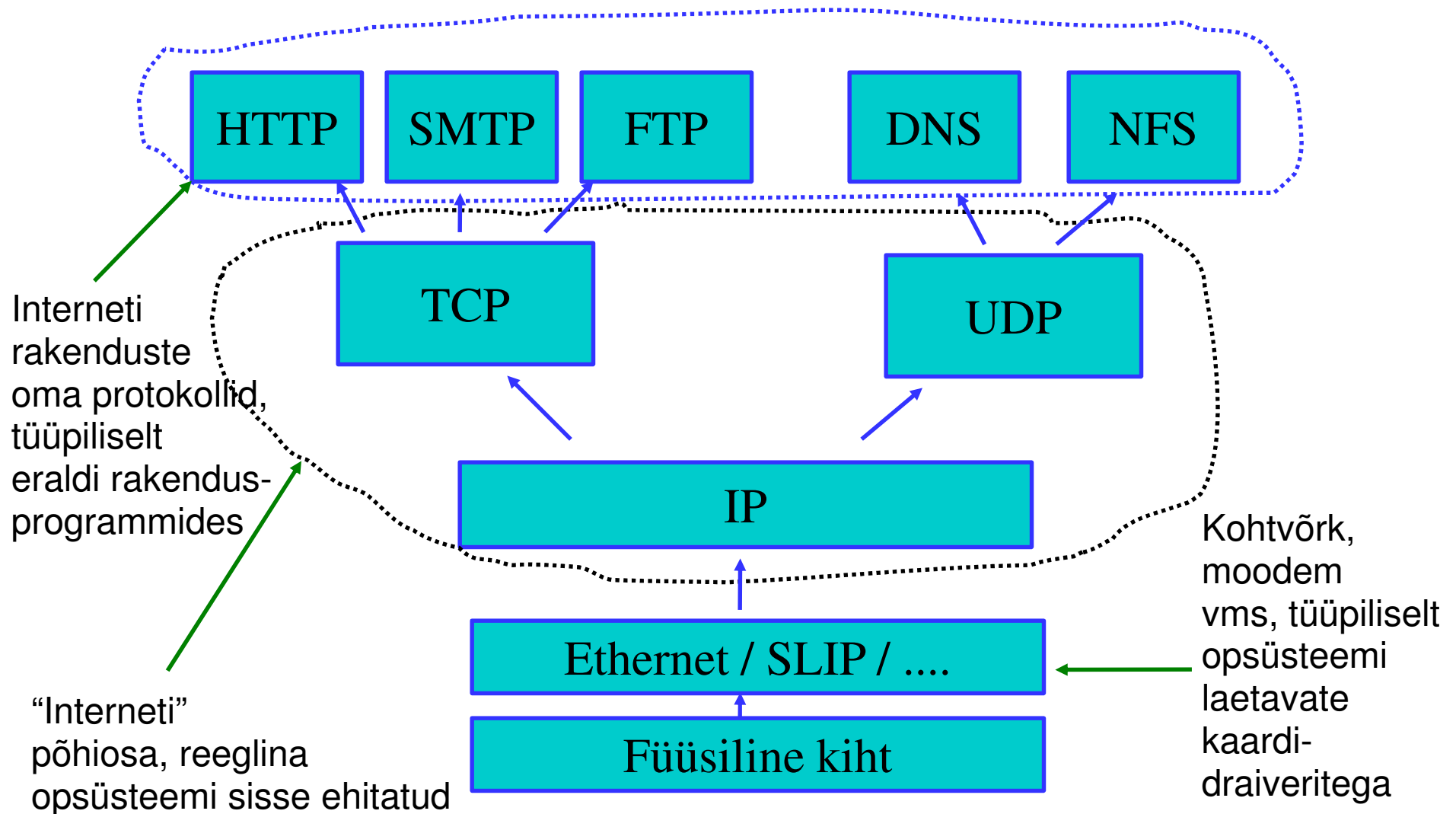
## Paketipõhised protokollid

nimetatakse ka *olekuta (stateless)* protokollideks  
suhtlus toimub paketi kaupa  
kommunikatsioon on “küsimus-vastus vormis”  
näiteks DNS, NFS, UDP, IP, Ethernet

# Protokollipered

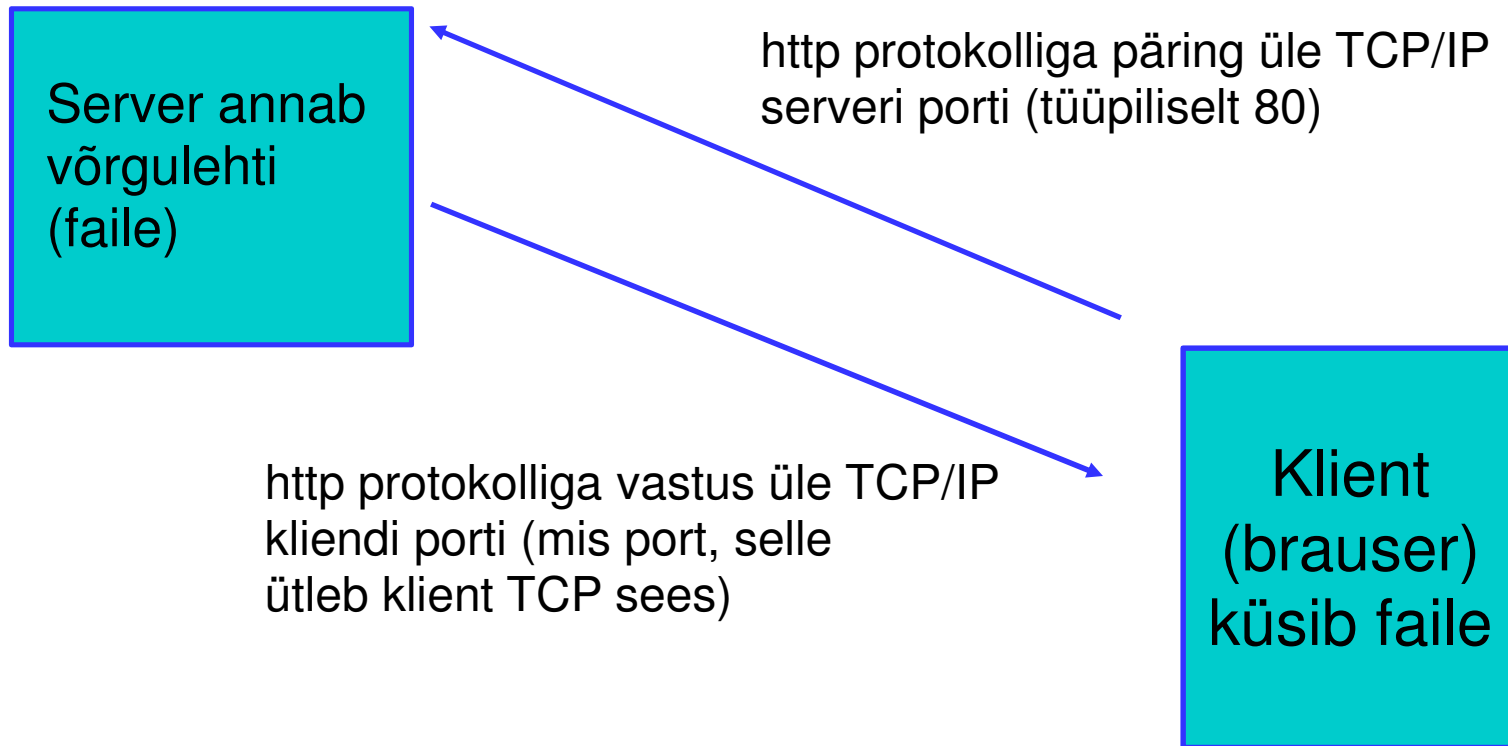


# Vahekokkuvõte protokollindusest internetis



# HTTP ühendused: failide küsimine ja nende andmine

- **HTTP on omaette protokoll**, mida kasutatakse veebilehtede, piltide, tekstifailide, zip failide jne jne saatmiseks veebiserveri ja brauseri vahel.



## Milline on HTTP protokollis päring?

- **HTTP päring on sisuliselt tekstiline käsk serverile:** “anna mulle selline fail”, kus näidatakse ära:
  - konkreetne küsimus-käsk
  - faili asukoht ja nimi
  - protokoll, mida küsija kasutab
  - ja soovi korral lisainfot, nagu küsija programmi tüüp (Mozilla, Ie, Konqueror, ..)
- **Näiteks:**
  - Võtame telnet ühenduse masina dijkstra.cs.ttu.ee porti 80
  - Tipime sisse, seejärel tipime kaks reavahetust:

```
GET index.html HTTP/1.0
```

## Milline on HTTP protokollis vastus?

- HTTP ei ole ehitatud “biti või baidi” tasemel, vaid teksti ridade kaupa: päis, tühi rida, tekstiread.

p  
ä  
i  
s

```
HTTP/1.1 200 OK
Date: Thu, 06 Nov 2003 13:50:07 GMT
Server: Apache/1.3.19 (Unix) PHP/4.1.1
Last-Modified: Sat, 10 Apr 1999 09:29:18 GMT
ETag: "46d8-297-370f19ee"
Accept-Ranges: bytes
Content-Length: 663
Connection: close
Content-Type: text/html
```

tühi  
rida

```
<html>
<head>
  <META HTTP-EQUIV="Content-type" CONTENT="text/html;
  charset=ISO-8859-1">
  . . . . .
```

Tegelik  
sisu

# Nimed Internetis

- Enamikul avalikel serveritel on DNS nimi
  - www.ttu.ee
  - google.com
- Serveril on IP aadress
  - 193.40.254.28 – neli täisarvu 0-255
  - 4294967296 unikaalset aadressi
  - See on seotud tema füüsilise asukohaga võrgus
  - Pakettide marsruutimine käib IP aadressi järgi
  - DNS nimede ja IP aadresside teisendust teeb DNS teenus
- IPv6 aadressid
  - 2001:0db8:85a3:0000:0000:8a2e:0370:7334
  - 340282366920938463463374607431768211456 aadressi
- Masinal on tavaliselt Etherneti (MAC )aadress
  - 01:23:45:67:89:ab
  - Unikaalne kohtvõrgu piires
  - Sõltub võrguliidesest



# DNS: Domain Name Server

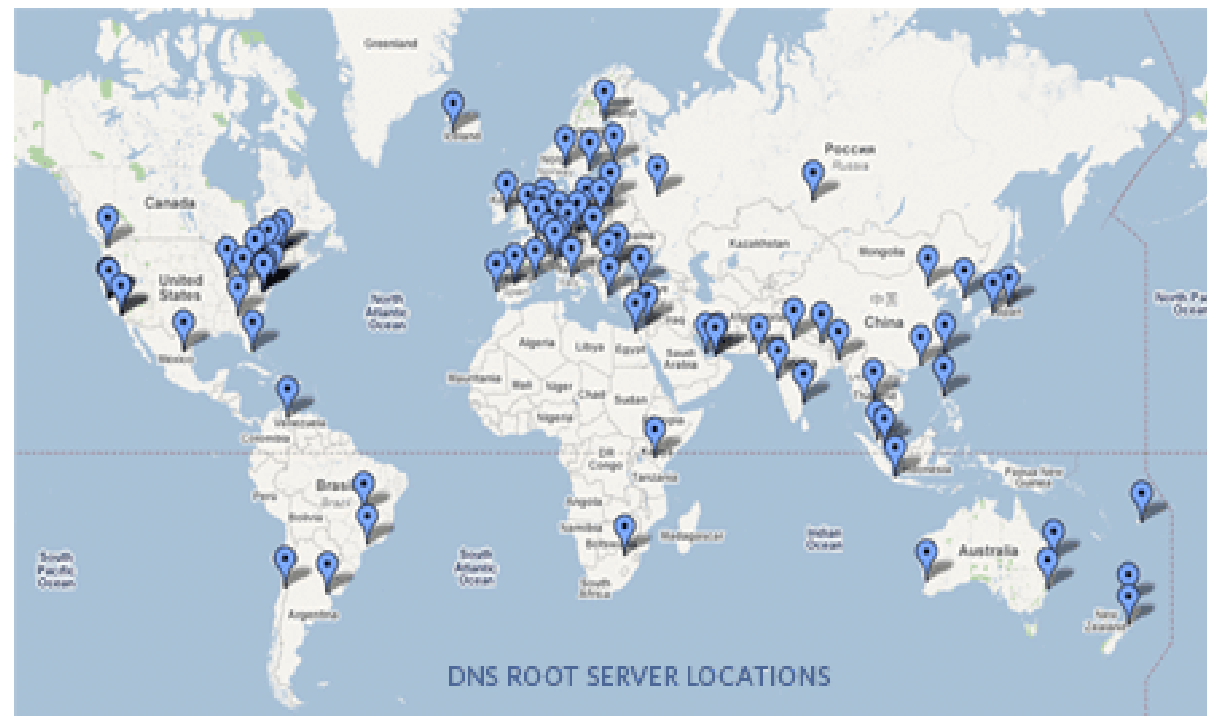
- **Ülesanne:** kui programmile on antud masina nimi, millega ühendust võtta (näiteks www.ttu.ee), siis tuleb kõigepealt leida sellenimelise masina IP aadress.
- **DNS serverid:** serverid, mis sisaldavad “nimi<->IP aadress” tabeleid ja vastavad päringutele “anna selle nime IP aadress”:

www.ttu.ee <--> 193.40.254.185

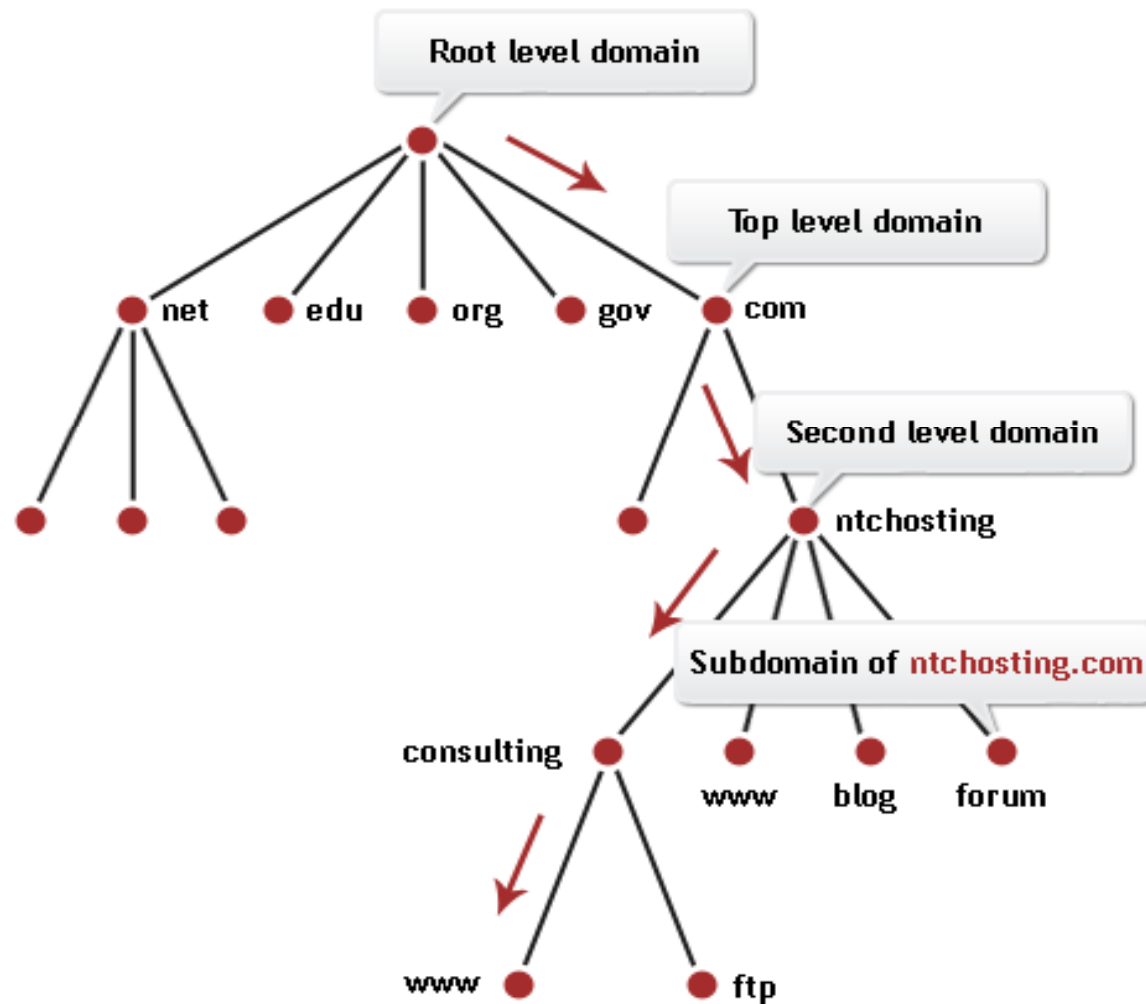
- Kuna DNS päring tuleb teha iga hariliku www-päringu jaoks, siis neid päringuid tehakse maailmas tohutult palju (rohkem kui mistahes muid infopäringuid). Üks masin või kanal ei pea seda vastu.
- Lahendus: **hajutatud andmebaas**. DNS servereid on maailmas väga palju. Igaüks saab üles panna oma DNS serveri. DNS serverid süngivad omavahel infot oma tabelites.

# DNS: Domain Name Server

- 2012 novembris oli registreeritud ca 200 000 000 domeeninime.
- Maailmas on ca 20 miljonit DNS serverit.
- Siin on “DNS hiearhia” ülemiste, nn root serverite asukohad:

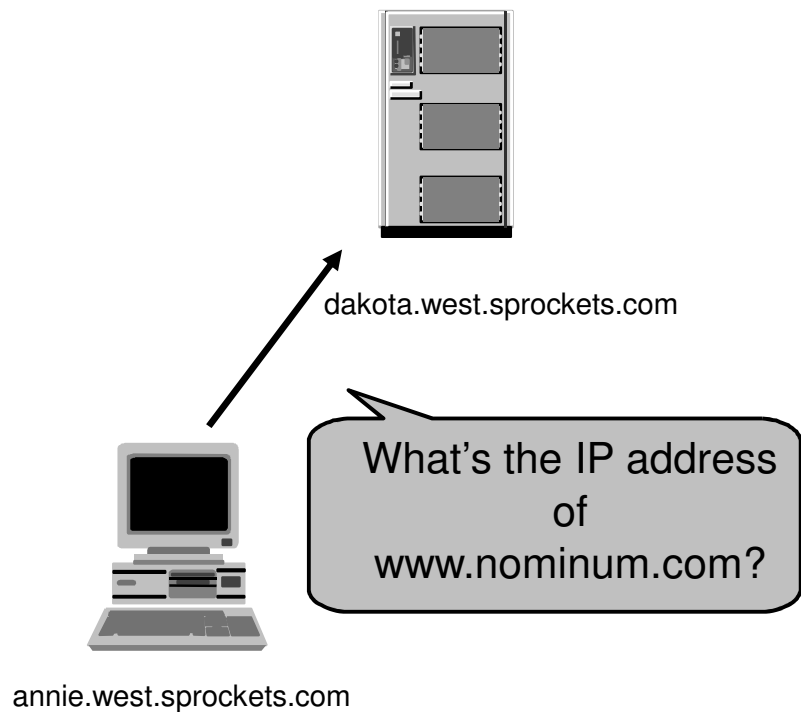


## DNS nime de puu



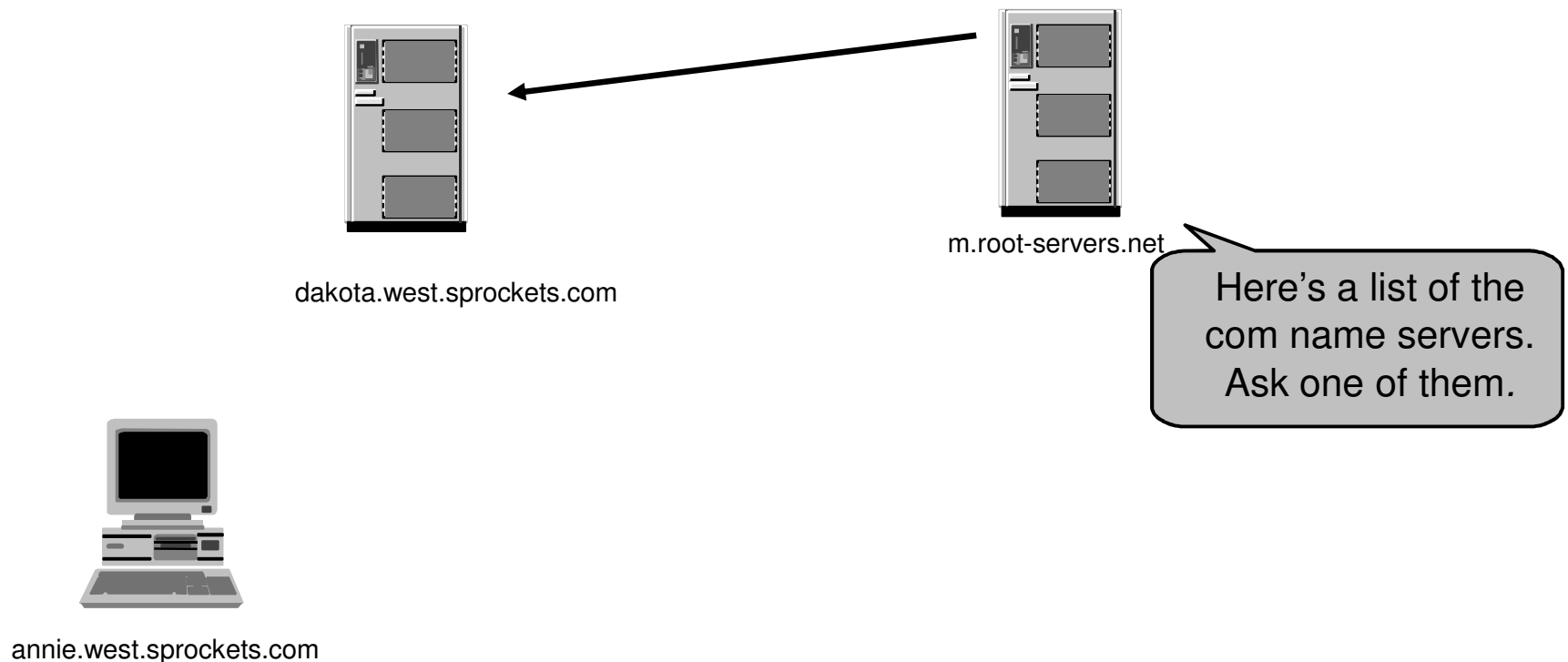
## DNS nimele lahendamise protsess

Arvuti *annie* oma nimeserverilt *dakota* aadressi  
*www.nominum.com*



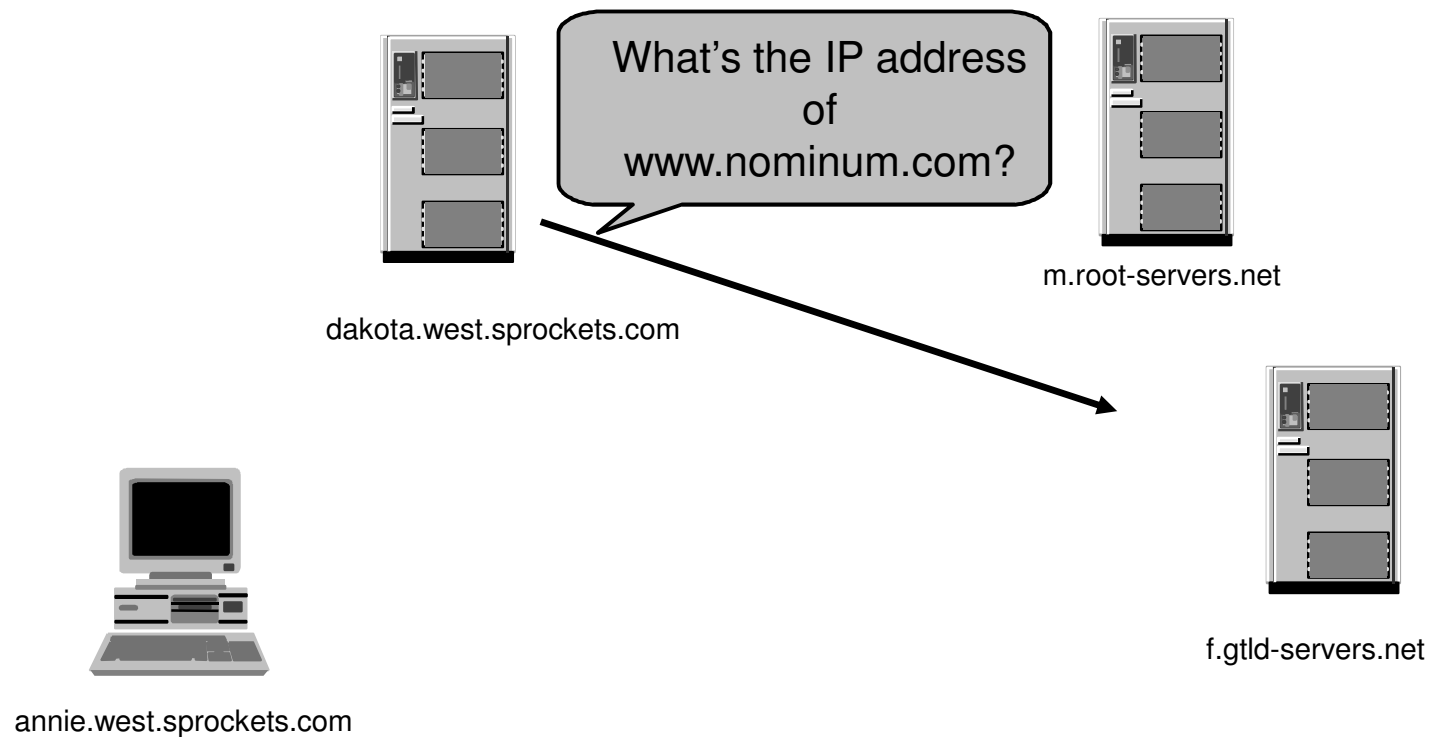
## DNS nimele lahendamise protsess

root server *m* annab *dakota*-le mis *.com* nimeserverid.  
Selline vastus on tüüpi “referral”



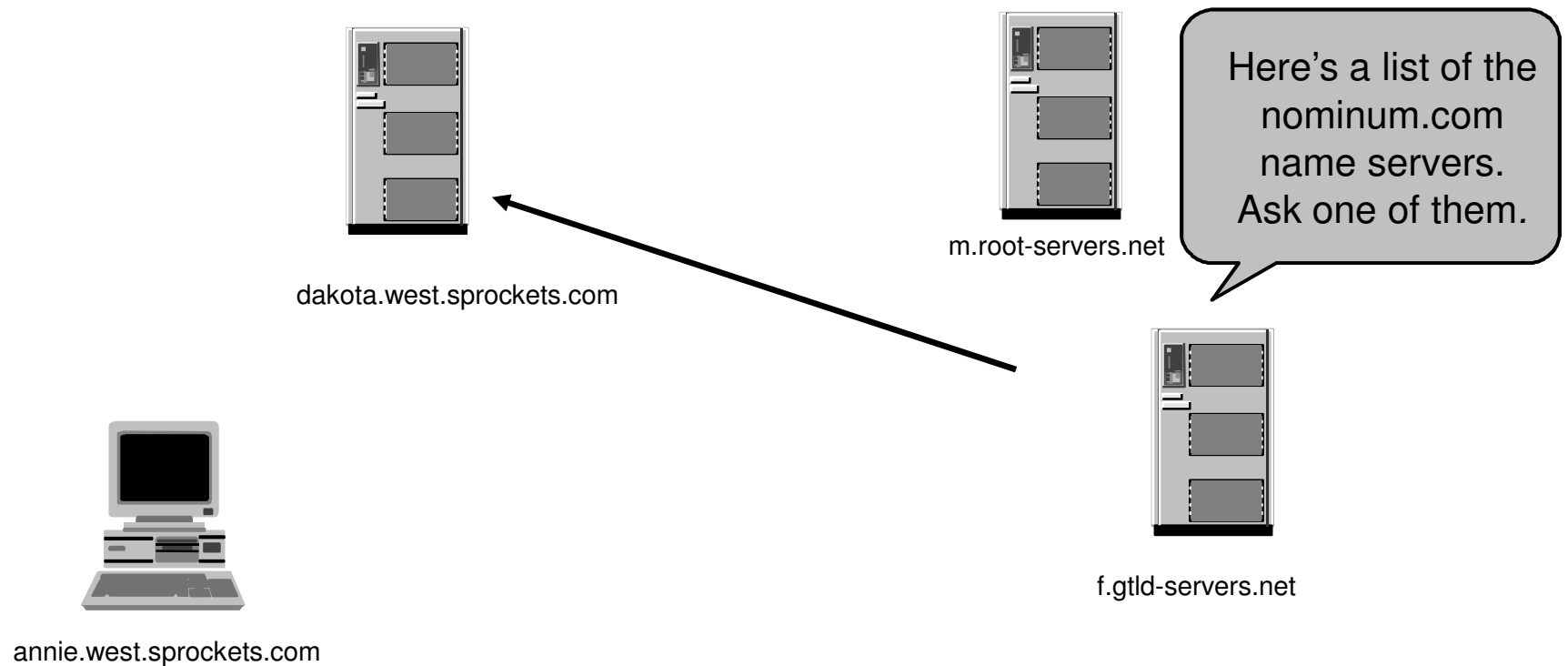
## DNS nimele lahendamise protsess

Nimeserver *dakota* küsib *.com* nimeserverilt *f* *www.nominum.com*'s aadressi



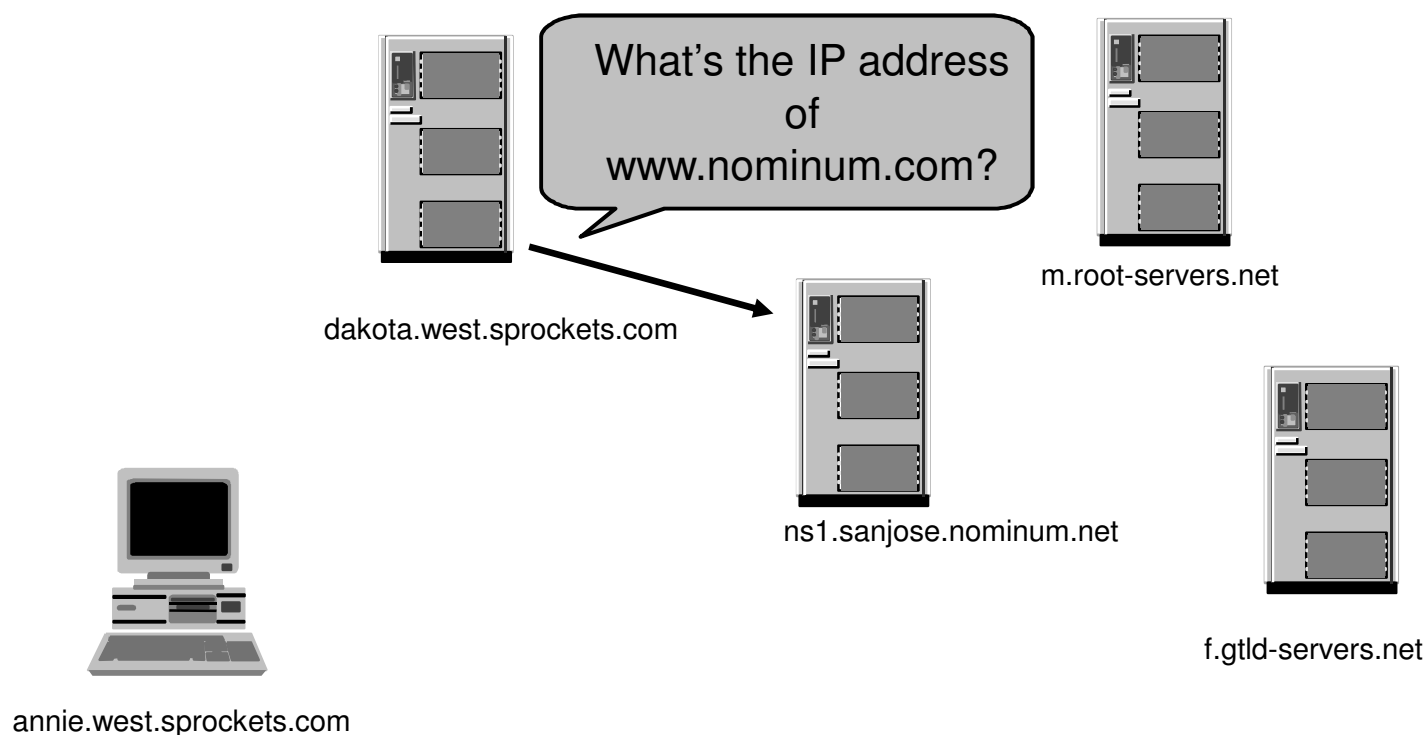
## DNS nimele lahendamise protsess

*.com* nimeserver *f* teatab *dakota*-le domeeni *nominum.com* nimeserverid



## DNS nimele lahendamise protsess

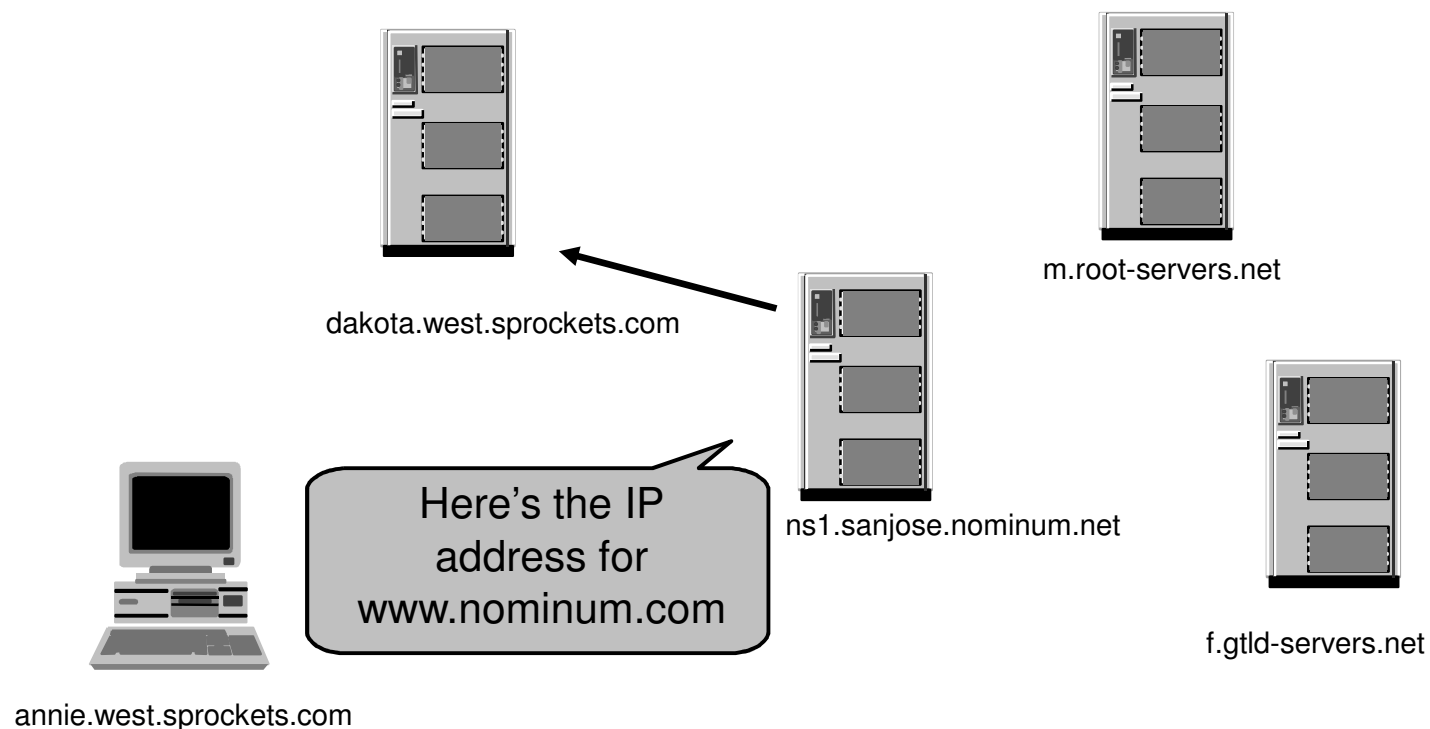
Nimeserver *dakota* küsib *nominum.com* nimeserverilt *ns1.sanjose* serveri *www.nominum.com*'s aadressi





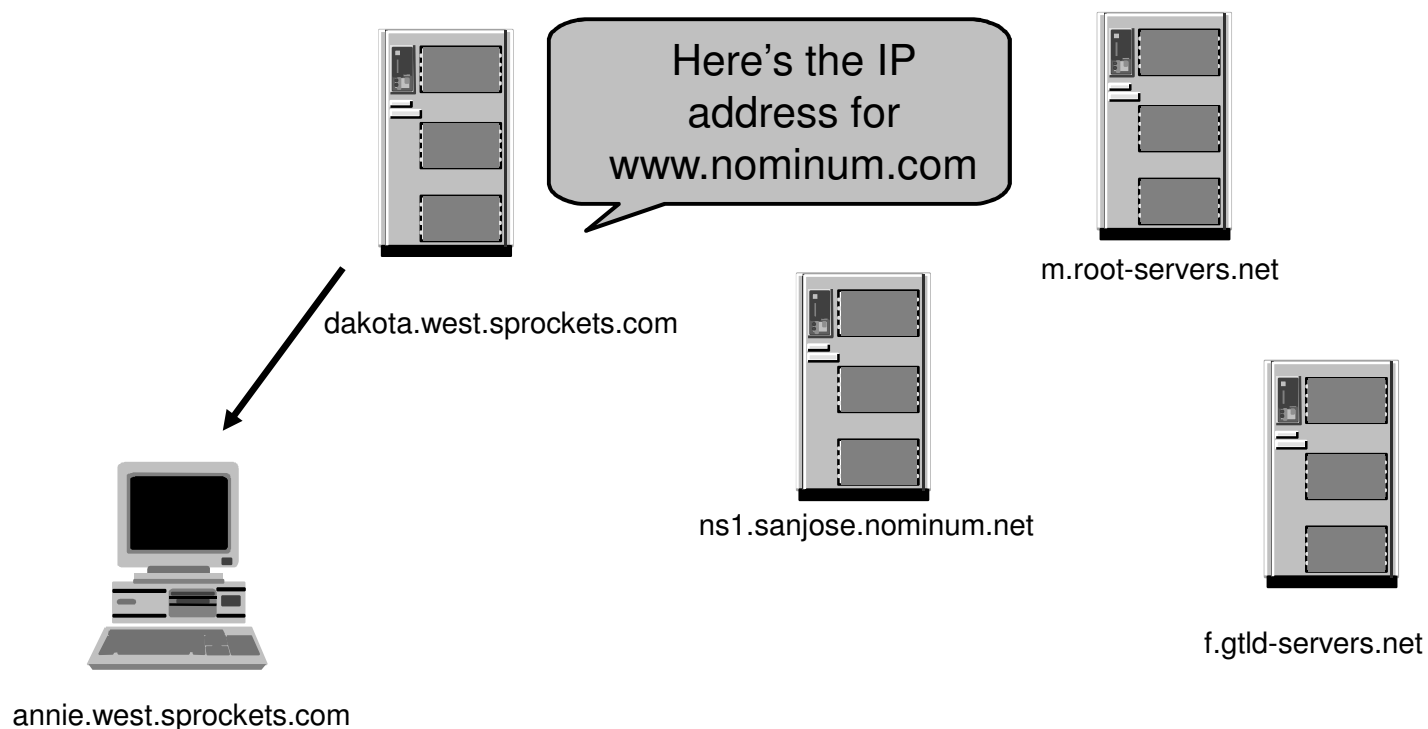
## DNS nimele lahendamise protsess

*nominum.com* nimeserver *ns1.sanjose* saadab vastuseks *www.nominum.com*'s aadressi



## DNS nimele lahendamise protsess

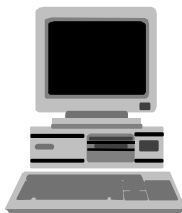
Nimeserver *dakota* vastab masinale *annie* ja saadab *www.nominum.com*'s aadressi



## DNS nime lahendamise protsess (puhverdamine)

Peale eelnevat päringut teab *dakota* nüüd ja jätab meelde:

- .com nimeserverite nimed ja IP aadressid
- e *nominum.com* nimeserverite nimed ja IP aadressid
- *www.nominum.com* IP aadressi



annie.west.sprockets.com

## Kogu protsess telnetist HTTP näitel (klient)

### ■ HTTP

- Annab välja faile serverarvutist ja veidi lisainfi (HTTP päis)
- Server asub TCP pordil 80

**% telnet www.ttu.ee 80**

- küsitakse nimeserverilt “www.ttu.ee” IP aadress
  - nimeserver teeb DNS protokolliga rekursiivse otsingu  
annab tulemuse 193.40.254.185
- telnet loob TCP ühenduse
  - reserveeritakse kohalik TCP port
  - saadetakse serverile TCP ühenduse avamise kutse

## Kogu protsess HTTP näitel (võrk)

- telnet saadab päringu
  - saadetakse TCP paketti sisaldav IP datagramm 193.40.254.185 le
- marsruutimistabelist selgub, et tuleb kasutada vaikimisi marsruutimisreeglit (Näiteks: 192.168.0.1)
- ARP abil tehakse kindlaks selle Etherneti aadress
  - saadetakse üldlevi aadressil küsimus 192.168.0.1
  - saadakse vastus 00:E0:1E:67:F4:3F
- saadetakse Etherneti kaader aadressil 00:E0:1E:67:F4:3F
  - oodatakse kuni Etherneti võrk on vaba
  - saadetakse kaader
- 192.168.0.1-s ja teistes marsruuterites korratakse marsruutimisprotseduuri

## Kogu protsess Http näitel (server)

- www.ttu.ee IP kiht annab datagrammi andmeosa edasi TCP kihile
- www.ttu.ee TCP kiht annab pordi numbri järgi andmeosa edasi http serveri-programmile
- http server-programm töötleb päringut
- http server-programm saadab vastuse ja lõpetab TCP ühenduse
  - kordub protseduur läbi TCP, IP ja Etherneti

ME, WHEN MY  
INTERNET ISN'T  
WORKING

WHAT AM I  
DOING WITH MY  
LIFE?

